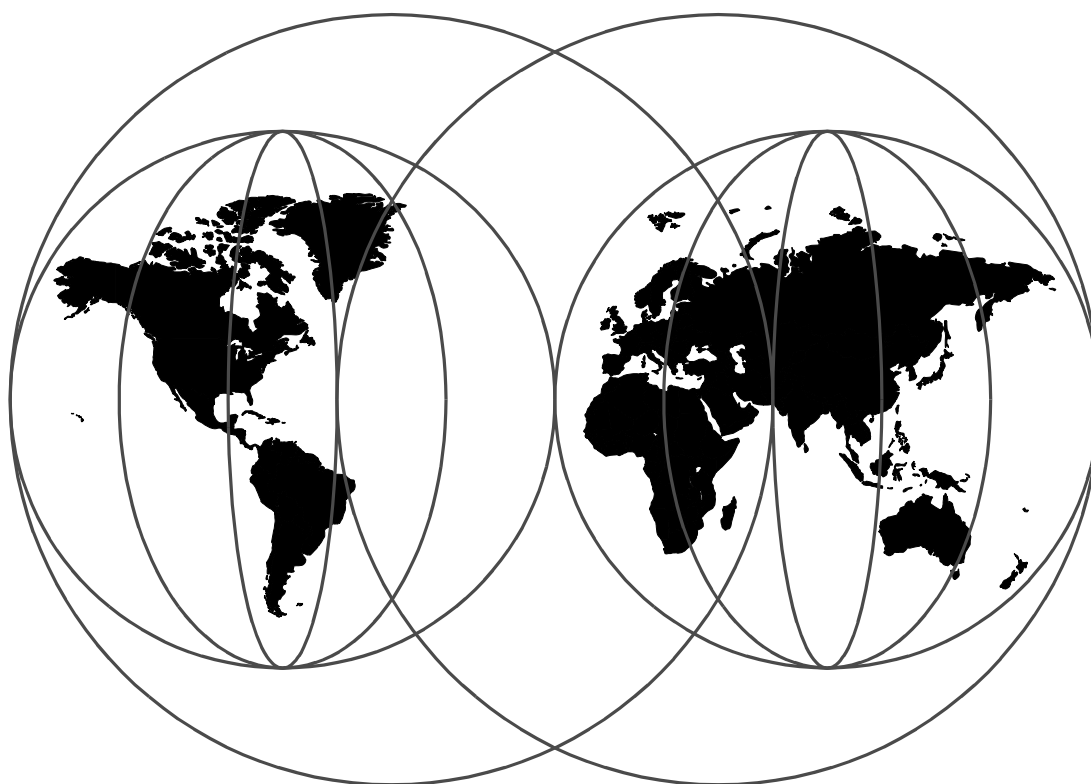


IBM WebSphere Performance Pack Usage and Administration

Marco Pistoia, Vincenzo Iovine, Stefano Pishedda



International Technical Support Organization

<http://www.redbooks.ibm.com>



International Technical Support Organization

SG24-5233-00

**IBM WebSphere Performance Pack
Usage and Administration**

December 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix C, "Special Notices" on page 423.

First Edition (December 1998)

This edition applies to Version 1.0 of IBM WebSphere Performance Pack for use with the AIX, Solaris and Windows NT operating systems.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HRBB Building 678
P.O. Box 12195
Research Triangle Park, NC 27709-2195

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1998. All rights reserved

Note to U.S Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	ix
Tables	xvii
Preface	xix
The Team That Wrote This Redbook	xix
Comments Welcome	xx

Part 1. IBM WebSphere Performance Pack Components 1

Chapter 1. Introduction to IBM WebSphere Performance Pack	3
1.1 The AFS File Sharing System in Distributed Computing Environments . . .	4
1.2 Caching and Filtering to Manage Internet Traffic and Bandwidth Demand. .	5
1.3 Load Balancing and Server Monitoring Capabilities	6
1.4 Building Record-Breaking Web Sites.	7
1.5 Who Can Benefit.	8
1.5.1 Content Hosting Internet Service Providers.	9
1.5.2 Corporate Web Sites and Content Aggregators.	10
1.5.3 Corporate Headquarters Buildings or Large Campuses.	12
1.5.4 Backbone Internet Service Providers.	12
1.5.5 Access Internet Service Providers.	13
1.5.6 Access ISPs with Subscriber Home Page Hosting.	14
1.6 Other IBM WebSphere Offerings.	15
1.6.1 IBM WebSphere Application Server	15
1.6.2 IBM WebSphere Studio.	16
Chapter 2. File Sharing Component	17
2.1 Features of the File Sharing Component	17
2.1.1 The Concept of AFS Cell.	17
2.1.2 Single File Name Space	17
2.1.3 AFS Volumes	18
2.1.4 Replication and Caching	18
2.1.5 Cached Data Validation.	19
2.1.6 Content Backup.	19
2.1.7 AFS Security	20
2.2 Why Do I Need an Enterprise File System?	20
2.2.1 AFS and NFS	20
2.3 How the File Sharing Component Works.	21
2.4 Installation and Configuration of the File Sharing Component	21
2.4.1 Installation of the First AFS Machine on AIX	22
2.4.2 Configuration of the First AFS Machine on AIX	29
2.4.3 Installation of a Stand-Alone AFS Client on AIX	52
2.4.4 Configuration of a Stand-Alone AFS Client on AIX	53
2.4.5 Installation of a Stand-Alone AFS Client on Windows NT	58
2.4.6 AFS Patch Installation on Windows NT	69
2.4.7 Configuration of a Stand-Alone AFS Client on Windows NT	72
2.5 Creation of an AFS User	85
2.5.1 Authenticating As admin	86
2.5.2 Creation of an Entry in the Protection Database	87
2.5.3 Creation of an Entry in the Authentication Database	88
2.5.4 User's Volume Creation	88

2.5.5	User's Volume Mount Point	88
2.5.6	Full Ownership and Access Rights on the Home Directory	89
2.5.7	Setting a Space Quota on the Volume	89
2.5.8	Setting an Appropriate Access Control List	89
2.5.9	AFS User Authentication	90
2.5.10	Experimenting with Access Control Lists	90
2.6	AFS Integrated Logon	91
2.6.1	AFS Integrated Logon on AIX	91
2.6.2	Interactions between the AFS Logon and the AIX Logon	97
2.6.3	AFS Integrated Logon on Windows NT	100
2.7	A Closer Look at AFS Cached Data Validation	103
Chapter 3. Caching and Filtering Component		
3.1	Features of the Caching and Filtering Component	108
3.2	Why Do I Need a Caching and Filtering Proxy Server?	109
3.2.1	Caching Proxy Function	109
3.2.2	Filtering Function	110
3.3	Cache Management	110
3.3.1	Controlling Which Documents Are Kept in the Cache	110
3.3.2	Cache Freshness	111
3.3.3	Cache Size and Garbage Collection	114
3.3.4	Cache Indexing	114
3.3.5	Automatic Cache Refreshing	115
3.4	PICS-Based Filtering	116
3.5	Flexible-Client SOCKS	117
3.6	Other Functions	119
3.6.1	Handling Header Information	119
3.6.2	Proxy Activity Monitor	119
3.6.3	Caching and Filtering Proxy Server Access Protection	119
3.6.4	Proxy Chaining	120
3.6.5	SSL Tunneling	120
3.7	Installation of the Caching and Filtering Component	120
3.7.1	Installation on AIX	120
3.7.2	Restarting the Caching and Filtering Proxy Server on AIX	128
3.7.3	Installation on Windows NT	129
3.7.4	Restarting the Caching and Filtering Proxy Server on Windows NT	147
3.8	Basic Configuration	149
3.8.1	Using the Configuration and Administration Forms	152
3.8.2	Enabling Proxy Function	153
3.8.3	Enabling the Pure Proxy Functionality	157
3.8.4	Enabling Basic Caching	158
3.8.5	Garbage Collection	162
3.9	Proxy Chaining Scenario	164
3.9.1	How to Set Up a Proxy Chaining Environment	166
3.9.2	Proxy Chaining Experiences	169
3.10	Proxy Server Protection	174
3.10.1	Resource Protection	175
3.10.2	Directives to Protect Access to the Proxy	175
3.10.3	Using the htadm Command to Manage Password Files	178
3.10.4	Testing the Configuration	178
3.11	Platform for Internet Content Selection Protocol	180
3.11.1	What Is the PICS Protocol?	181
3.11.2	PICS Content Labels	181

3.11.3	Rating Systems	181
3.11.4	Rating Services	182
3.11.5	PICS Filtering at the Proxy Server Level	184
3.12	PICS Scenario	186
3.12.1	Network Configuration	187
3.12.2	Configuration of the Web Server	188
3.12.3	Configuration of the Rating Service	188
3.12.4	Configuration of the Label Bureau	194
3.12.5	PICS Filter Configuration on the Proxy Server	198
3.12.6	Working with PICS Filtering at the Proxy Server Level	201
3.12.7	Embedding the PICS Labels in the Web Documents	204
Chapter 4.	Load Balancing Component	209
4.1	Functions of the Load Balancing Component	210
4.1.1	Dispatcher	210
4.1.2	Interactive Session Support	211
4.2	Why Do I Need the Load Balancing Component?	211
4.3	How the Dispatcher Function Works	213
4.3.1	Dispatcher Components	214
4.3.2	Proportions of Importance	215
4.3.3	Information Flow	216
4.3.4	TCP Ports Used by the Dispatcher	217
4.4	How the ISS Function Works	218
4.4.1	ISS Cells and Services	218
4.4.2	ISS Configuration Files	219
4.4.3	Cell and Its Attributes	222
4.4.4	Nodes	223
4.4.5	Services	224
4.4.6	Resources	224
4.4.7	Metrics	225
4.4.8	ISS Observers	226
4.4.9	ISS Selection Methods	228
4.5	Installation of the Load Balancing Component	229
4.5.1	Installation on AIX	229
4.5.2	Installation on Windows NT	235
4.6	Load Balancing Basic Scenario Scenario Using the Dispatcher	246
4.6.1	Network Environment	246
4.6.2	Cluster Address and Non-Forwarding Address	247
4.6.3	Dispatcher Configuration	248
4.6.4	TCP Servers Configuration	267
4.6.5	How the Dispatcher Works: The Flow of the IP Packets	273
4.6.6	Round-Robin Load Balancing Scenario	275
4.6.7	Analyzing the Flow with a Network Monitoring Tool	280
4.6.8	Activating the Manager and the Advisors	282
4.6.9	Customization of the Manager and the Advisors	285
4.6.10	Saving the Configuration	287
4.7	Installation of the Interactive Session Support Function	290
4.7.1	Installation on AIX	291
4.7.2	Installation on Windows NT	292
4.8	Load Balancing Scenario Using the Dispatcher and ISS Functions	294
4.8.1	ISS Configuration File	295
4.8.2	Managing ISS	298
4.9	Rule-Based Load Balancing	300

4.9.1 How Are Rules Evaluated?	302
4.10 Rule-Based Load Balancing Scenario	302
4.10.1 Network Environment.	303
4.10.2 Rules Configuration	304
4.11 Dispatcher High Availability	313
4.12 Dispatcher High Availability Scenario	314
4.12.1 Network Architecture	314
4.12.2 Configuration Steps Common on Both the Machines	316
4.12.3 Configuration Steps for High Availability	320
4.12.4 Experimenting with High Availability.	332
4.13 Firewall High Availability Using the Load Balancing Component	338
4.13.1 Basic Configuration Issues.	339
4.13.2 Setting the Rules for IBM eNetwork Firewall	342
4.13.3 Scenario Implementation	344

Part 2. IBM WebSphere Performance Pack Scenarios 347

Chapter 5. Enterprise Campus Scenario	349
5.1 First Step: Traditional Scenario	350
5.2 Second Step: Enterprise Campus Complete Scenario	355

Chapter 6. Regional and Local Access ISP Scenario	367
6.1 First Step: Regional and Local Access ISP Basic Scenario	370
6.2 Second Step: Regional and Local Access ISP Complete Scenario	379

Chapter 7. Large ISP Scenario	399
--	------------

Appendix A. Network Configuration of the Scenario Environments.	405
A.1 Computer Environment Configuration	405
A.2 IP Configuration	406
A.2.1 External LAN	406
A.2.2 Internal LAN	408
A.2.3 IP Router	409
A.3 TCP/IP and Adapter Tuning	409
A.3.1 Maximum Transmission Unit	409
A.3.2 The tcp_sendspace and tcp_recvspace Parameters.	409
A.3.3 The MBUFS Parameter	410
A.4 Increasing the Performances on the Web Client Machine.	410
A.4.1 Number of Processes per User	410
A.4.2 Paging Space.	410

Appendix B. Scenario Basic Configuration	411
B.1 Installation of the Client Platform	411
B.2 Web Servers Configuration to Support WebStone's Workload	412
B.3 Configuration of the Client Platform	412
B.3.1 Creation of the File /rhosts	413
B.3.2 Disabling DNS	413
B.3.3 WebStone Configuration Files	413
B.3.4 Launching WebStone.	416
B.4 Information Provided by WebStone	417
B.5 Scenario Example	418

Appendix C. Special Notices	423
Appendix D. Related Publications	425
D.1 International Technical Support Organization Publications.	425
D.2 Redbooks on CD-ROMs	425
D.3 Other Publications.	425
How to Get ITSO Redbooks	427
How IBM Employees Can Get ITSO Redbooks	427
How Customers Can Get ITSO Redbooks	428
IBM Redbook Order Form	429
Index	431
ITSO Redbook Evaluation	435

Figures

1. How to Implement a WebSphere Performance Pack Environment	7
2. Content Hosting Internet Service Providers	9
3. Corporate Web Sites and Content Aggregators	10
4. Head Office - Branch Office	11
5. Hierarchical Caching	13
6. Access Internet Service Providers	14
7. Load Balancing Component Language Selection	24
8. Choose Destination Location	24
9. Select the File Sharing Server and File Sharing Client Components	25
10. Enlarge the Size of the File System /	26
11. Choose to Not Replace Version	27
12. Start Installation Window	27
13. Installation Complete	28
14. Verifying AFS Installation	29
15. Add Logical Volume for AFS Partition	30
16. Create a Standard Journaled File System	31
17. Mounting the Journaled File System JFS over the Directory /vicepa	32
18. Update of afsConfigureServer.ksh	34
19. Prompt for AFS and admin User Password	35
20. Output of the afsConfigureServer.ksh Command	35
21. Status of the AFS Server Processes	41
22. Checking if the User admin Has Been Successfully Created	41
23. Contents of the File /usr/vice/etc/dkload/rc.afsd.large	44
24. Contents of the File /usr/vice/etc/dkload/rc.afsd.medium	45
25. Contents of the File /usr/vice/etc/dkload/rc.afsd.small	45
26. Contents of the File /etc/rc.afs in Our AFS Client and Server AIX Machine	50
27. AFS Server Processes Displayed by the ps -ef Command	51
28. AFS Client Processes on the First AFS Machine	52
29. Installing a Stand-Alone AFS Client on AIX	53
30. Contents of the File /etc/rc.afs on the Stand-Alone AFS Client AIX Machine	56
31. AFS Client Processes on the Stand-ALone AFS Client	57
32. Virtual Memory Configuration on Our Windows NT AFS Client Machine	58
33. Installation Program's Name and Path	59
34. Selection of Java Virtual Machine Version	59
35. Load Balancing Component Language Selection	60
36. Choose Destination Location	61
37. Selection of File Sharing Client component	62
38. Replacing the Current Version of Any Product Already Installed	62
39. AFS Client Setup Complete	63
40. The Runtime Installation Execution Has Failed	64
41. Runtime Installation Execution Error Window	64
42. AFS Client Configuration Panel - Client Properties	66
43. AFS Client Configuration Message	66
44. AFS Configuration Client Panel - Cell Host Choices	67
45. Add Cell Panel while Entering Data	68
46. Add Cell Panel after Data Have Been Accepted	68
47. Installation Complete	69
48. Transarc AFS Patch	70
49. Information Panel	70
50. A First Warning Message During the Installation of the Patch 9	71

51. Second Warning Message during the Installation of the Patch 9	71
52. Setup Complete Panel	72
53. Windows NT Task Manager As Soon As the Cache Manger Starts	73
54. Transarc AFS Daemon Startup Type	74
55. Control If AFS Client Process Is Running	75
56. How To Mount the AFS File System	76
57. Accessing the AFS File System	77
58. Configuration Panel netdate	78
59. Advanced Configuration Panel	79
60. AFS Authentication Panel	81
61. Tokens Panel	82
62. Authentication Complete	82
63. Output of the tokens Command	83
64. AFS Menu in the File Manager Window	84
65. AFS Pop-Up Menu	85
66. Error Message Displayed When Trying to Access a Protected File	91
67. Particulars of the afsuser UNIX User	92
68. Authenticating on AIX but Not on AFS	99
69. AFS Configuration Client Panel	100
70. Creation of the Windows NT User afsuser	101
71. Error Message If the AFS Daemon Has Not Started	102
72. Authentication Panel after Reboot	102
73. Delving Process	116
74. Error 403 with an HTML Document Blocked by Filtering Rule	117
75. Traditional Proxy Server	118
76. Flexible-Client SOCKS in the Caching and Filtering Component	119
77. Load Balancing Component Language Selection	122
78. Choose Destination Location	123
79. No Space Available for Installation	124
80. Free Space Enough for the Installation	125
81. Choose to Replace Version	125
82. Start Installation Window	126
83. Installation Complete	126
84. Checking Whether the httpd Daemon Is Running	127
85. Installation Program's Name and Path	129
86. Selection of Java Virtual Machine Version	130
87. Choose Option If No JVM Is Installed	130
88. Load Balancing Component Language Selection	131
89. Choose Destination Location	132
90. Forcing the System to Create the Destination Location	132
91. Selecting the Caching and Filtering Component	133
92. Choose to Replace Version	134
93. Installation Problem Window	135
94. Runtime Installation Execution Error	135
95. Installation Command	136
96. Choose Target Directory	137
97. Enabling or Disabling the Java Servlet Support	137
98. Problems with the Installation	138
99. Stopping the Setup Process	139
100. IBM WebTraffic Express among the Available Services	140
101. Select Component Window	141
102. Choose Directory for Installation	142
103. Deciding Where to Install Each Item	142

104.Deciding Where to Install Each Item	143
105.Reinstalling the httpd.cnf Configuration File	144
106.Reinstalling the ics_pics.cnf Configuration File	144
107.Reinstalling the javelin.cnf Configuration File	144
108.Reinstalling the socks.cnf Configuration File	145
109.Reinstalling the admin.pwd Configuration File.	145
110.Configuration Values for Web traffic Express	145
111.Enabling or Disabling the Java Servlet Support	146
112.Setup of the Caching and Filtering Component is Complete.	147
113.Starting the IBM Web Traffic Express Service.	147
114.IBM Web Traffic Express Service Window	148
115.IBM Web Traffic Express Graphical User Interface.	149
116.Caching and Filtering Component Front Page	151
117.Entering Administrator User Name and Password	151
118.Proxy Configuration abd Administration Form.	152
119.Proxy Server Setting	153
120.Confirmation Page.	156
121.Restart Confirmation Page	157
122.Performance Settings	158
123.Caching Setting Form	159
124.Confirmation Window.	161
125.Cache Storage Reuse	163
126.Cache Storage Reuse Confirmation Page.	164
127.Graphical Representation of the Proxy Chaining.	165
128.Architecture of the Proxy Chaining Environment.	166
129.Proxy Chaining and Non-Proxy Domains Form.	168
130.Proxy Chaining Confirmation Page	169
131.venus Cache Access Log File - 1	170
132.rs600030 Httpd Log File - 1	170
133.rs600022 Cache Access Log File - 1.	171
134.rs600022 Cache Access Log File - 2.	171
135.venus Cache Access Log File - 2	172
136.rs600030 Httpd Log File - 2	172
137.rs600022 Cache Access Log File - 3.	173
138.rs600030 Httpd Log File - 4	173
139.rs600022 Cache Access Log File - 5.	174
140.Entering a Wrong User ID and Password	179
141.Proxy Authentication Failed.	179
142.Entering a Valid User ID and Password	180
143.Confirmation of the Successful Authentication	180
144.PICS Filtering at the Proxy Server Level	185
145.PICS Scenario Environment	187
146.(Part 1 of 2). coolness.rat File	190
147.(Part 2 of 2). coolness.rat File	191
148.Web Page Pointed by the Rating System URL Identifier.	191
149.Web Page Pointed by the Rating Service URL Identifier.	192
150.age1.lbl File	193
151.age2.lbl File	193
152.age3.lbl File	193
153.age4.lbl File	193
154.age1.lbl File - Complete Version	194
155.ics_pics.conf File	196
156.javelin.cnf File	198

157.Demonstration of the failURL Directive	202
158.Another Demonstration of the failURL Directive	202
159.Demonstration of the passURL Directive	203
160.The Conditional Filter Passes the URL	203
161.The Conditional Filter Rule Blocks age3.html.	204
162.The Conditional Filter Rule Blocks age4.html.	204
163.PICS Environment without the Label Bureau Server	205
164.testage4.html	206
165.testage4.html Displayed on a Web Browser.	206
166.javelin.cnf - PICS Filtering	207
167.Error Message Displayed on the Web Browser	208
168.Information Flow Used in the Dispatcher	216
169.Cell, Services and Nodes	218
170.(Part 1 of 2). Iss_config_1 Sample Configuration File	221
171.(Part 2 of 2). Iss_config_1 Sample Configuration File	222
172.Load Balancing Component Language Selection	230
173.Choose Destination Location	231
174.No Space Available for Installation.	232
175.Enlarge the Size of the File System	233
176.Free Space Enough for the Installation	233
177.Choose to Replace Version	234
178.Start Installation Window	234
179.Installation Complete	235
180.Installation Program's Name and Path	236
181.Selection of Java Virtual Machine Version	236
182.Choose Option If No JVM Is Installed.	237
183.Load Balancing Component Language Selection	238
184.Choose Destination Location	239
185.Select Component Window	240
186.Choose to Replace Version	241
187.Installation Problem Window	242
188.Runtime Installation Execution Error	242
189.Installation Command.	243
190.Selection of Installation Type	243
191.Selection of Components	244
192.Select Language Files Window	244
193.Available Services on Windows NT	245
194.Execution Error.	245
195.Graphical Representation of the Scenario Environment	247
196.Server Component of the Dispatcher	250
197.Graphical User Interface of the Dispatcher Configuration	251
198.Start Executor.	252
199.Executor Status Window	253
200.ifconfig Command Entered to Discover the netmask Value	256
201.How to Verify That the Alias Has Been Created Correctly	256
202.Script File goldle.sample on AIX	258
203.Adding a Cluster	259
204.Enter the Cluster Address	259
205.Cluster Added.	260
206.Adding a Port	262
207.Enter the Port Number	262
208.Adding a Server	263
209.Enter Server Address.	264

210.Verifying That the Server Has Been Added	265
211.All Servers Added	266
212.Select Network Adapter: MS Loopback Adapter	268
213.MS Loopback Adapter Card Setup	268
214.Adapters List Refreshed	269
215.Configuring MS Loopback Adapter	270
216.Network Setting Change - Reboot Your System	270
217.Extra Route	271
218.Extra Route Deleted	272
219.Startup Folder of Windows NT Server	272
220.TCP Connections on Port 80	276
221.D:\WWWHTML\enzo.html on Host 9.24.104.44	276
222.D:\WWWHTML\enzo.html on Host 9.24.104.224	277
223./usr/lpp/internet/server_root/pub/enzo.html on Host 9.24.104.127	277
224.HTML Page Served by the First Web Server	277
225.The First Web Server Has Served One Page	278
226.HTML Page Server by the Second Web Server	278
227.The Second Web Server Has Served One Page	279
228.HTML Page Served by the Third Server	279
229.The Third Web Server Has Served One Page	280
230.Using the Network Monitor	281
231.Starting the Manager	282
232.How to Start the Advisor	283
233.Select Advisor	284
234.Advisor Added	284
235.Other Advisors Added	285
236.Setting the Manager Proportions and Smoothing Level	287
237.Enter Configuration File Name	288
238.Configuration File eNDconf01	289
239.Loading a Configuration File	290
240.Installing ISS on AIX	291
241.Selecting ISS Components	292
242.Select Language Files Window	293
243.IBM_ISS_Load_Balancing Service Available on Windows NT	294
244.Graphical Representation of the Scenario Environment	295
245.ISS Configuration File Issbase.conf	297
246.IBM_ISS_Load_Balancing Service on Windows NT	299
247.Graphical Representation of the Rule-Based Load Balancing Scenario	304
248.Basic Dispatcher Configuration	305
249.Dispatcher GUI Monitor	306
250.Adding a Rule	307
251.Add a Rule Panel	308
252.The Configuration Has Been Updated	310
253.Adding a Server to a Rule	311
254.Entering the Server's IP Address	311
255.Updated Rule Status Window	312
256.Verifying That the Rule Configuration Has Been Successful	313
257.High-Availability Scenario Configuration	315
258.Backpup Dispatcher - Executor Status	317
259.Backpup Dispatcher - Cluster Status	318
260.Backpup Dispatcher - Port Status	319
261.Backpup Dispatcher - Manager Status	320
262.Backpup Dispatcher - Add Backup Panel	321

263.Primary Dispatcher - Add Backup Panel	322
264.Primary and Backup Dispatcher - Add a Reach Target	323
265.Backup Dispatcher - High Availability Status	324
266.Primary Dispatcher - High Availability Status	325
267.Checking for Heartbeat.	326
268.Network Interface Status - Backup Machine	326
269.Network Interface Status - Primary Machine	327
270.goActive Script	328
271.goStandby Script	329
272.goInOp script	329
273.Save Configuration File - Backup Machine	330
274.Load Configuration File - Backup Machine.	330
275.Network Interface Status - Backup Machine	331
276.Network Interface Status - Primary Machine	332
277.High Availability Status Window	333
278.Takeover Disabled When the Recovery Strategy Is Set to Auto	334
279.Takeover Enabled When the Recovery Strategy Is Set to Manual	335
280.Takeover Disabled on the Primary Dispatcher Machine	336
281.Takeover Confirmation Window	337
282.eND GUI on the Backup Machine after the Takeover	337
283.Takeover Enabled on the Primary Dispatcher Machine	338
284.Graphical Representation of the Scenario	340
285.goActive Script	341
286.goInOp script	341
287.goStandby script.	342
288.Enterprise Campus Scenario	349
289.Graphical Representaion of the Traditional Scenario.	351
290.WebStone testbed File	352
291.WebStone filelist File	352
292.Basic Scenario - Web Alone - Connection Rate Average	353
293.Basic Scenario - Web Alone - Throughput Average for All Connections	353
294.Basic Scenario - Web Alone - Average Latency.	354
295.Windows NT Task Manager on the Web Server Machine	355
296.Flow Diagram of the Enterprise Campus Scenario	356
297.Dispatcher Configuration - Executor Status	358
298.Dispatcher Configuration - Cluster Status	359
299.Dispatcher Configuration - Port Status	360
300.Dispatcher Configuration - Manager Status	361
301.WebStone testbed File	362
302.WebStone filelist File	362
303.httpd-log File on the Web Server rs600030	363
304.httpd-log File on the Web Server wtr05195	364
305.httpd-log File on the Web Server computer1	364
306.Scenario 1 - Connection Rate Average	365
307.Scenario 1 - Throughput Average for All Connection	365
308.Scenario 1 - Average Latency	366
309.Regional and Local Access ISP Scenario Architecture	367
310.Flow Diagram of the Regional and Local Access ISP Basic Scenario	371
311.testbed File	373
312.WebStone filelist File	374
313.Active Connections 70 Clients	375
314.New Connections 70 Clients	376
315.Servers' Weights 70 Clients	377

316.Basic Scenario - Connection Rate Average	378
317.Basic Scenario - Throughput Average for All Connections	378
318.Basic Scenario- Average Latency	379
319.Regional and Local Access ISP Flow Diagram	380
320.Proxy Dispatcher Configuration - Executor Status	383
321.Proxy Dispatcher Configuration - Cluster Status	384
322.Proxy Dispatcher Configuration - Port Status	385
323.Proxy Dispatcher Configuration - Manager Status	386
324.WebStone testbed File	387
325.WebStone filelist File	388
326.Active Connections 100 Clients	389
327.New Connections 100 Clients	390
328.Servers' Weights 100 Clients.	391
329.Active Connection's 100 Clients	392
330.New Connection's 100 Clients.	393
331.Servers' Weights 100 Clients.	394
332.Connection Rate Average	395
333.Throughput Average for All Connections.	395
334.Average Latency	396
335.Large ISP Scenario Architecture	399
336.Flow Diagram of the Large ISP Scenario	401
337.Display the Routing Tables on AIX Machine	407
338.Host Name Resolution File for Computers in the Internal LAN	408
339./home/guest/WebStone-1.1/conf/filelist/filelist.dynamic-light	414
340.Modified File /home/guest/WebStone-1.1/conf/filelist/filelist.dynamic-light . .	415
341.Configuration File home/guest/WebStone-1.1/conf/testbed	416
342.Graphical Representation of the Scenario Flow	418
343.WebStone Filelist.	419
344.testbed Configuration File	420
345.HTML Sample WebStone Report	421
346.Basic Scenario - Throughput Average for All Connections	421
347.Basic Scenario - Average Latency.	422
348.Basic Scenario - Connection Rate Average	422

Tables

1. The Particulars of Our AFS Environment	104
2. Environment Configuration for Proxy Chaining	167
3. PICS Scenario - Hardware, Software and Network Configuration	187
4. Basic Scenario - Hardware, Software and Network Configuration	246
5. Protocol and Port Used by the Dispatcher	261
6. Extra Route	271
7. Advisors and Related Protocol and Ports	282
8. Rule-Based Scenario - Hardware, Software and Network Configuration	303
9. High Availability Scenario - Hardware, Software and Network Configuration	314
10. Basic Configuration	340
11. High Availability Settings	344
12. Traditional Scenario - Hardware, Software and Network Configuration	351
13. Complete Scenario - Hardware, Software and Network Configuration	357
14. Basic Scenario - Hardware, Software and Network Configuration	372
15. Complete Scenario - Hardware, Software and Network Configuration	381
16. Large ISP Scenario - Hardware, Software and Network Configuration	402
17. Internal LAN - Hardware, Software and Network Configuration	405
18. Router Machine - Hardware, Software and Network Configuration	405
19. External LAN - Hardware, Software and Network Configuration	406
20. Software Products Used in the Scenarios	406
21. Hardware, Software and Network Configuration	418

Preface

IBM WebSphere Performance Pack is Web infrastructure software that addresses the scalability, reliability and performance needs of e-business applications in both local and geographically distributed environments. Its functions incorporate leading-edge and robust caching, file management and load balancing, that together compensate for the inherent weakness of the Internet to support critical business applications and expectations.

This redbook will give you a clear understanding of the features of IBM WebSphere Performance Pack. It shows how to plan for, install, configure, use, tune and troubleshoot each component and offers specific implementation examples. It helps explain how to build complex scenarios that involve all the components of IBM WebSphere Performance Pack, to give you a better understanding of the technologies involved.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

The leader of this project was Marco Pistoia.

Marco Pistoia is an International Technical Support representative working as a project leader at the International Technical Support Organization, Raleigh Center. He writes extensively and teaches IBM classes worldwide on all areas of the Network Computing Framework, Java and Internet security. Marco holds a degree with honors in Pure Mathematics from the University of Rome and a masters degree in Computer Science. Before joining the ITSO, he was a System Engineer in IBM SEMEA Sud, Italy. He received an Outstanding Technical Achievement Award in 1996.

Vincenzo Iovine is a System Engineer in IBM SEMEA Sud, e-business Solutions Center of Naples, Italy. He holds an Electronic Engineering degree, Telecommunication specialization, from the University of Napoli. He joined IBM six years ago. He has three years of experience in the multimedia applications project and development, and three years of experience in network computing. His areas of expertise include Web solution designs. He has designed and developed a number of Web-based applications, specifically using Lotus Domino and IBM Web servers.

Stefano Pishedda is a System Engineer in IBM SEMEA Sud, e-business Competence Center of Cagliari, Italy. He holds an Electrical Engineer degree from the University of Cagliari. He has worked with IBM for three years. His areas of expertise include AIX (RS/6000, SP Systems) and Windows NT in Internet and intranet environments, with particular focus on Internet security and firewalls.

Thanks to the following people for their invaluable contributions to this project:

Susan Hanis, Joan Cavin, Steve Roma, Barbara Kemper, Lisa Tomita,
Rick Schenck, Blake Corbitt, Jeremy Noonan
IBM Research Triangle Park, North Carolina

Bob McNamara
IBM, Integrated Solutions Marketing, Somers

Margaret Ticknor
IBM, International Technical Support Organization, Raleigh

Mary Ann DelBusso, Catherine Milligan
Transarc Corporation

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in "ITSO Redbook Evaluation" on page 435 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

For Internet users <http://www.redbooks.ibm.com>

For IBM Intranet users <http://w3.itso.ibm.com>

- Send us a note at the following address:

redbook@us.ibm.com

Part 1. IBM WebSphere Performance Pack Components

Chapter 1. Introduction to IBM WebSphere Performance Pack

IBM WebSphere Performance Pack is a Web infrastructure software that addresses the scalability, reliability and performance needs of e-business applications in both local and geographically distributed environments. Its functions incorporate leading-edge and robust caching, file management and load balancing, that together compensate for the inherent weakness of the Internet to support critical business applications and expectations.

IBM WebSphere Performance Pack has been developed using IBM's extensive experience with very demanding Web sites.

IBM WebSphere Performance Pack is built up of three main components, which permit you to reduce Web server congestion, increase content availability and improve Web server general performances:

1. File Sharing

The File Sharing component, also known as the AFS file system, is an enterprise file system that enables co-operating hosts (client and servers) to efficiently share file system resources across both local area networks and wide area networks. It provides non-disruptive real-time replication of information across multiple servers, which guarantees data consistency, availability, global stability and administrative efficiency, required by large distributed Web sites or by Web sites with volatile content requiring considerable administrative effort to maintain content links and URLs to file I/O mapping.

2. Caching and Filtering

The Caching and Filtering component, also known as Web Traffic Express (WTE), is a caching proxy server that provides highly scalable caching and filtering functions associated with receiving requests and serving URLs. With tunable caching capable of supporting high cache hit rates, this component can reduce bandwidth costs and provide more consistent rapid customer response times.

3. Load Balancing

The Load Balancing component, also known as eNetwork Dispatcher (eND), is a server that is able to dynamically monitor and balance TCP servers and applications in real time. The main advantage of the Load Balancing component is that it allows heavily accessed Web sites to increase capacity, since multiple TCP servers can be dynamically linked in a single entity that appears in the network as a single logical server.

The above components, which were previously unavailable in a single Internet software offering, can increase the scalability, availability and reliability of your Web site, while reducing infrastructure costs.

Installation procedures permit selection of which components to install, and specification of which machine(s) the selected component(s) should be located. Subject to installation needs and operating platforms, components can coexist on a single machine or can be distributed over multiple machines.

IBM WebSphere Performance Pack Version 1.0 is supported on the following platforms:

- Any IBM RISC/6000-based machine running IBM AIX 4.2.1 or later
- Any SPARC workstation supported by Sun Solaris 2.5 or later
- Any Intel x86 PC supported by Microsoft Windows NT 4.0.

Before you begin your installation plans, you should consider the following:

- The File Sharing server component is supported only on UNIX systems, not on Windows NT.
- The File Sharing server component cannot be installed on the same workstation with the Caching and Filtering or Load Balancing components.
- On Windows NT, for load balancing purposes, the Load Balancing component cannot be installed on the same workstation with the Caching and Filtering component.

1.1 The AFS File Sharing System in Distributed Computing Environments

AFS has provided scalable file administration and file sharing for large enterprises for many years, based upon its use of a virtual name space to make naming and logical directory structures of files independent of their physical location. AFS clients and AFS servers are used to establish this virtual name space capability. In typical LAN file systems this is achieved by installing AFS clients on user workstations, communicating with an AFS server that manages the I/O operations associated with the actual files. In a Web site, the AFS clients can be installed on HTTP servers to reduce the administrative effort associated with maintaining URL to file I/O mapping relationships. In addition, HTTP servers that are simultaneously AFS clients can significantly increase the connectivity capacity to Web server content and can provide local and geographically distributed access efficiency.

AFS is a central and scalable file system:

- It is *central* because AFS brings together all of the files within the file system into a single name space. Every AFS user shares this same name space, making all AFS files easily available from any AFS machines. With AFS, the name of a file is independent of both the file's and the user's physical location, contributing to ease of file sharing and resource management.
- It is *scalable* because AFS is able to manage a very large number of files, spread across many geographical locations. When remote files, residing on AFS servers, are accessed by remote AFS clients, they are cached on the client machines to improve performance. This makes remote working across global distances feasible, since it is possible to access your own files from sites many thousands of miles away as if they were local.

Both small and large-scale distributed environments benefit from AFS mechanisms to reduce server and network load:

- AFS caches data on client machines to reduce subsequent data requests directed at file servers, substantially reducing network and server loads. Servers keep track through *callbacks* of data given to clients, guaranteeing cache consistency without constant queries to the server to see if the file has changed. It is important to underline also that AFS allows disk cache, not just memory cache. This is a key advantage of AFS over other shared file systems.

- The AFS remote procedure call (RPC) reads and writes data to an RPC stream, further improving the efficiency of data transfer across a local or wide area network.

Extended security is guaranteed through Kerberos authentication and access control lists (ACLs). AFS Kerberos-based authentication requires that users prove their identities before accessing network services. Once authenticated, AFS access control lists give individual users or groups of users varying levels of permission to perform operations on the files in a directory.

As we see in Chapter 2, “File Sharing Component” on page 17, the File Sharing component offers also replication techniques for file system reliability. Multiple copies of frequently accessed (but infrequently changed) data are replicated on multiple file servers within a cell. When accessing this information, a client will choose among the available servers that house replicas. If one server is unavailable or unreachable, the client will go to another server. Replication also reduces the load on any particular server by placing frequently accessed information on multiple servers.

Moreover, management utilities are provided to ease the load of system administrators in growing environments. Backup, reconfiguration and routine maintenance are all done without any system down time. Files remain available to users during these operations. This is done by creating online clones of volumes.

AFS commands are RPC-based. Administrative commands can be issued by any authenticated administrator from any client workstation. System databases track data location information, authentication information and protection groups. These databases are replicated on multiple servers, and are dynamically updated as information changes. Server processes accomplish many tasks automatically, such as restarting servers, tracking file locations and updating file servers with new binaries and configuration files.

1.2 Caching and Filtering to Manage Internet Traffic and Bandwidth Demand

The Caching and Filtering component of IBM WebSphere Performance Pack is both a caching proxy server and a content filter. The advanced caching of this component minimizes network bandwidth and ensures that end users spend less time when retrieving the same content multiple times.

These components act as gateways for multiple clients and perform basic Web server duties, such as receiving requests and serving URLs.

A traditional proxy server receives a request for an URL from a client and it forwards the request to the destination content server. The Caching and Filtering component of IBM WebSphere Performance Pack does something more; it can save or cache Web documents it retrieves and serve subsequent requests for those documents from its local cache. The client gets the requested information faster and network bandwidth is reduced.

As we see in detail in Chapter 3, “Caching and Filtering Component” on page 107, this component of IBM WebSphere Performance Pack also offers other key features of advanced caching, such as:

- The ability to handle very large caches

- An option to automatically refresh the cache with the most frequently accessed pages
- The possibility to cache even those pages where the header information says to fetch them every time
- Configurable daily garbage collection, to improve server performance and ensure cache maintenance.

The Caching and Filtering component of IBM WebSphere Performance Pack allows you to set content filtering at the proxy server level, rather than or in addition to the browser level, where it could be easily compromised or over-ridden.

Content filtering in the Caching and Filtering component of IBM WebSphere Performance Pack can use:

- Platform for Internet Content Selection (PICS) rules guiding use of rating labels - such as Recreational Software Advisory Council on the Internet (RSACi) criteria for inappropriate language, nudity or violence - placed in HTML or HTTP headers or third-party content rating label distributions
- Lists of URLs/sites for which access is to be blocked
- APIs for filtering applications

1.3 Load Balancing and Server Monitoring Capabilities

The Internet has grown so rapidly over the last few years, you are probably looking for a way to handle your company's share of that traffic. If this growth is not properly handled, users get slow response or refused connections, creating an unsatisfactory user experience which may cause the user never to visit your site again. Internet sites can become unstable or even fail under critical load conditions. What is needed is a solution that balances the load effectively and protects the user from these bad experiences.

IBM eNetwork Dispatcher has been developed to address these limitations and provide customers with advanced functions to meet their site's scalability and availability needs. It consists of two functions: the Dispatcher and the Interactive Session Support (ISS), which we describe in detail in Chapter 4, "Load Balancing Component" on page 209.

These two components can be deployed separately or together in various configurations to suit a wide variety of customer application requirements:

- You can use the Dispatcher function to balance the load on the server within a local area network or wide area network using a number of weights and measurements that are set dynamically.
- You can use the ISS function to balance the load on servers within a local area network or wide area network using a domain name server round-robin approach or a more advanced user-specified approach. ISS periodically monitors the level of activity on a group of servers and detects which server is the least heavily loaded.
- Using both components together allows you to balance the load on servers within both local and remote networks.

Both the components offer a built-in high availability feature:

- The Dispatcher high availability feature involves the use of a secondary machine that monitors the main, or primary, machine and stands by to take over the task of load balancing, should the primary machine fail at any time. This feature is available on all the platforms where IBM WebSphere Performance pack is supported, without using High Availability Cluster Multi-Processing (HACMP).
- In ISS high availability, all the nodes in a site work together to eliminate any single point of failure.

As we see in 4.13, "Firewall High Availability Using the Load Balancing Component" on page 338, the high availability feature provided by the Dispatcher function can be successfully used even in other configurations, for example to guarantee firewall high availability.

1.4 Building Record-Breaking Web Sites

IBM WebSphere Performance Pack allows you to design several architectures to enhance the performance of your Web site. The following figure offers an idea of the multiple configurations that can be obtained combining the WebSphere Performance Pack components:

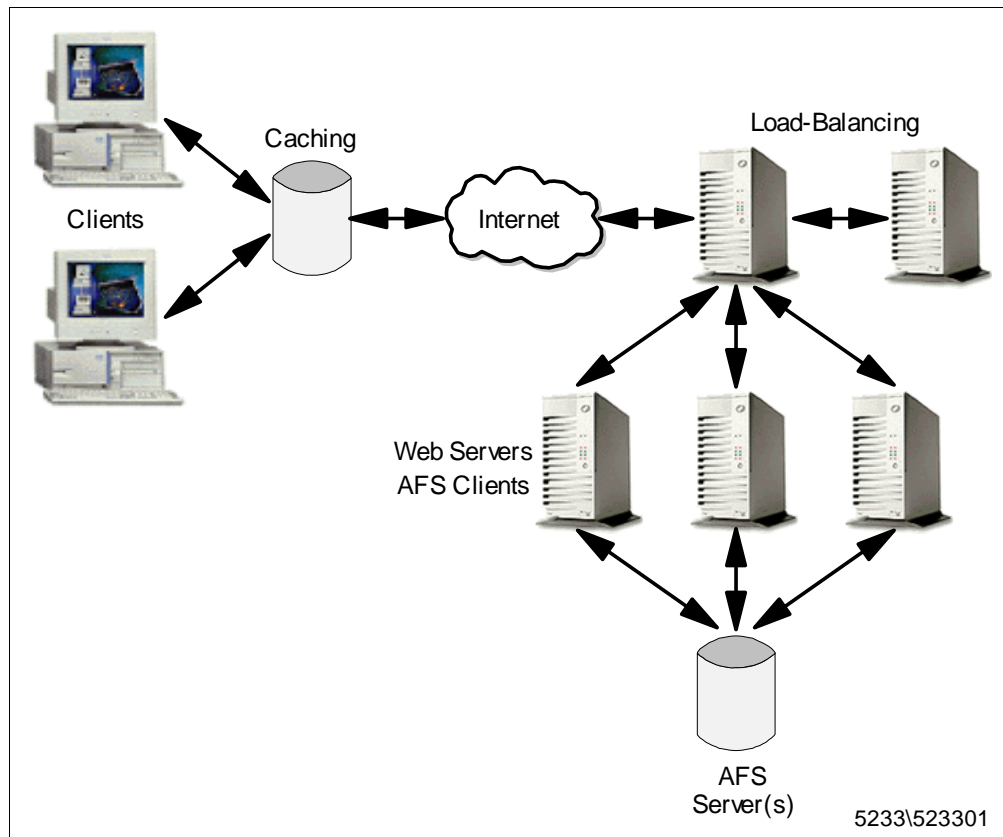


Figure 1. How to Implement a WebSphere Performance Pack Environment

- The Caching and Filtering component minimizes response time and network bandwidth utilization by providing Web content caching. It also ensures reliable content filtering at the proxy server level.
- The Load Balancing component distributes the load between multiple clustered Web servers. As soon as a client request arrives, the load balancing machine uses sophisticated monitoring tools and then forwards the request to the least loaded server. High availability is provided by a backup Dispatcher machine, which monitors the state of the primary machine and takes over the primary machine if this fails.
- The clustered Web servers can share the same content by using the File Sharing component of IBM WebSphere Performance Pack. This ensures high scalability, reliable access to replicated data and an efficient security model for access and group management.
- The Web server selected by the load balancing machine can then respond directly to the client without any further involvement of the load balancing machine. Since there is no need for the server response to go back through the same physical path, a separate high-bandwidth connection can be used, such as Asynchronous Transfer Mode (ATM) and Fiber Distributed Data Interface (FDDI).

IBM used the WebSphere Performance Pack technology to create a scalable and reliable system that efficiently handled unprecedented traffic volumes. On February 17th, 1998, at 12:41 (Japan Standard Time), the Official Web site of the Olympic Winter Games in Nagano made Internet history by logging a staggering 98,226 hits per minute. Less than a week later, a new all-time record was established with a peak load of more than 103,400 hits per minute, while still providing normal response time. The Internet site of the 1998 Nagano Olympic Winter Games is recognized by the Guinness Book of World Records.

In addition to the Winter Olympics site, IBM has built some of the largest Web sites in the world. For example, IBM hosted the Deep Blue chess match, the 1996 Olympic Games, the U.S. Open tennis tournament, Wimbledon, the Masters golf tournament, and the official Web site of the recently concluded 1998 French Open tennis championship.

1.5 Who Can Benefit

IBM WebSphere Performance Pack allows you to design and perform multiple architectures. The scenarios described in this section provide specific examples of how various ISP implementations can benefit from the use of IBM WebSphere Performance Pack. In Part 2, “IBM WebSphere Performance Pack Scenarios” on page 347, we see all the details on how to implement the architectures described in this section.

Notice that IBM has a complementary caching offering, Distributed Web Traffic Express (DWTE), which uses the same code base as Web Traffic Express, the Caching and Filtering component of IBM WebSphere Performance Pack, but has specific feature enhancements. The complementary function used by DWTE is called Remote Caching Access (RCA) and offers support for reducing redundancy of cached content.

In the following, we show how some of the scenarios below may also need the functions of remote caching to reduce redundant page storage.

1.5.1 Content Hosting Internet Service Providers

Content hosting service providers can effectively use all the components of WebSphere Performance Pack to more effectively support and distribute the content from their own Web sites, and to provide more response and cost-effective access to other sites.

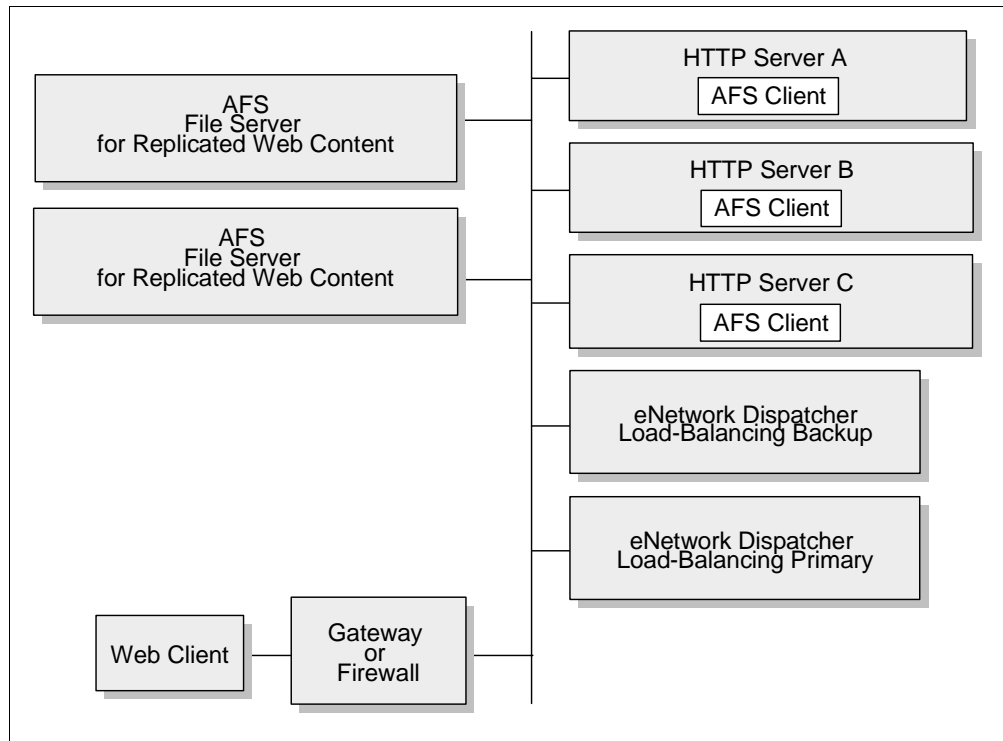


Figure 2. Content Hosting Internet Service Providers

Within a content hosting *server farm* the components can be configured to provide high availability and accessibility as follows:

- Content for hosted Web sites can be distributed over multiple volumes using the AFS virtual name space to simplify administration of page content. Because all of the Web servers need to have equal access to the Web content, a shared file system is an obvious choice for manageability of the Web content. AFS is a superior file system in this environment because of its replication capabilities, which provide improved availability and scalability. The disk caching capability of the AFS client ensures that network accesses to the file server are minimized.
- Scalability of access to the content can be achieved by adding multiple HTTP servers that are simultaneously AFS clients to access AFS server content, and by using efficient Load Balancing to dispatch requests to the HTTP server with the best capacity to handle the requests. The Load Balancing component of IBM WebSphere Performance Pack provides improved availability and scalability by allowing a farm of Web servers to provide a single Web site image to clients.

- High availability can be maintained by using AFS file replication capabilities, and by configuring a hot standby Load Balancing component.
- Proxy caching can be used to provide more responsive access to content from other sites, as well as to optimize backbone network traffic capacity.
- Both availability and performance may be further enhanced by geographically distributing HTTP servers with AFS clients, together with proxy caching for other internet content closer to user access points, such as points of presence (POPs) and network access points (NAPs).

Thus content hosting service provider or corporate webmasters can benefit from the non-disruptive replication and distribution capabilities of the File Sharing functions, local and wide area load balancing, and proxy caching. Flexible configuration of these components can ensure that requests are directed to the most appropriate local or remote location, and can enable location outages or routine maintenance schedules to be handled without disrupting customers.

The IBM WebSphere Performance Pack components can be used in conjunction with firewalls and authentication gateways to provide secure access where desired, and the load balancing function of WebSphere Performance Pack can also be used to scale these capabilities.

1.5.2 Corporate Web Sites and Content Aggregators

Use of IBM WebSphere Performance Pack by corporate Web sites and content aggregators is similar to that of Content Hosting Internet Service Providers.

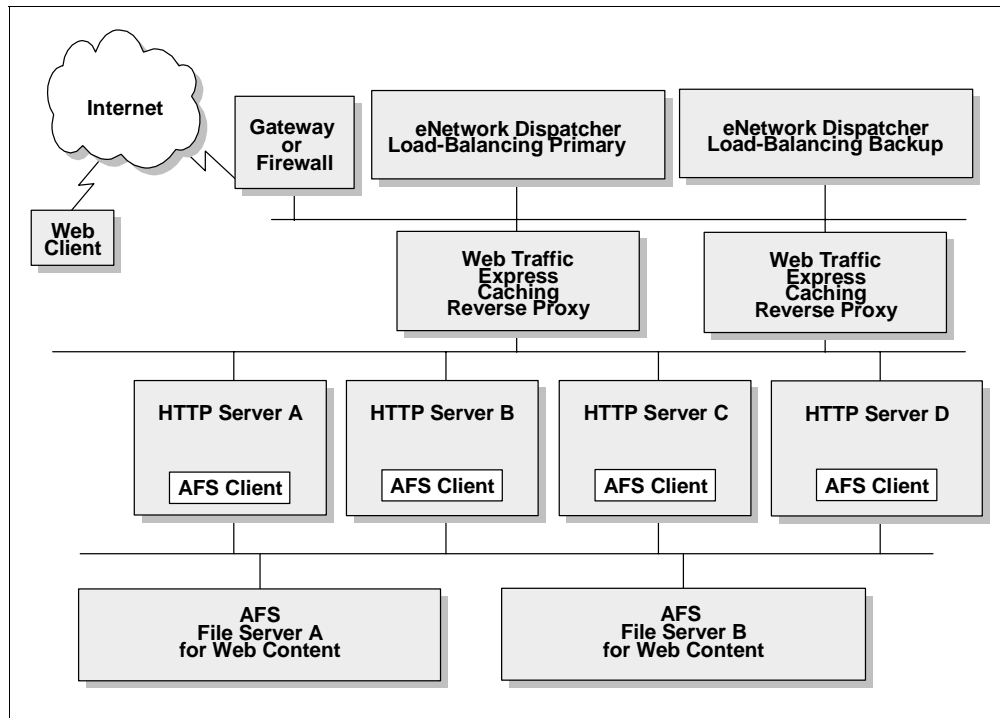


Figure 3. Corporate Web Sites and Content Aggregators

In many cases, corporate Web sites and content aggregators need to maintain a demilitarized zone (DMZ) to ensure that access to Web content by employees, business partners, or customers does not expose internal computer resources to

unauthorized users or hackers. However, firewalls cannot be deployed between AFS clients and their servers. In such instances, AFS replication can be used to establish read-only AFS servers within the DMZ. Here multiple AFS clients and the load balancing function can be used to provide the degree of scalability necessary to satisfy users. In addition, caching and filtering proxy servers may be deployed on the same machines or on separate systems to filter or optimize access from within the corporation to external Web sites.

Many corporate Web sites are located at head office or regional locations, while branch offices or business partners may need frequent access to the content. Most offices of this type have relatively low line speed (somewhat less than 1.5 Mbps) network access to the regional or corporate sites. Relatively few users can, with concurrent usage, use all the available bandwidth with resulting erratic response times. At these locations HTTP servers with AFS clients combined with general purpose caching can provide more consistent user response time.

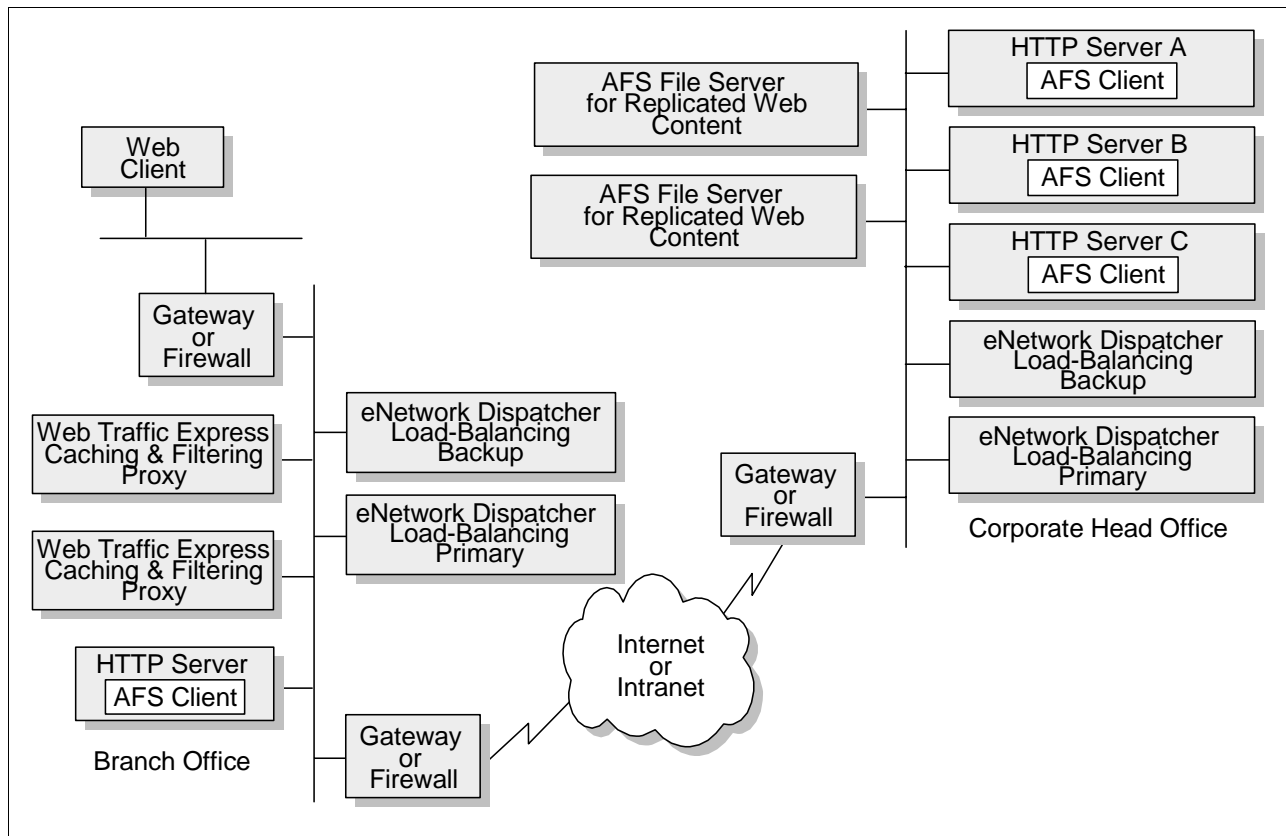


Figure 4. Head Office - Branch Office

Some industries have both small branch offices in addition to larger branch or regional offices. A simple deployment of IBM WebSphere Performance Pack caching and filtering on a single platform, eventually combined with a firewall, is probably sufficient to provide more consistent response time for employees in these offices. In the larger branches or regional offices, it may be desirable to have more than one caching and filtering proxy server and to use the Load Balancing functionality to provide better resource management.

Two approaches can be used to reduce redundant page storage and to address the effectiveness of caching in these locations:

1. Hierarchical caching

Optimize a primary (initial) cache for higher page hit rate by favoring more files of a smaller size, and a hierarchical cache for higher hit byte rate by favoring fewer files of a larger size.

2. Distributed Web Traffic Express

Configure RCA together with the Load Balancing function and with AFS shared file storage to reduce the index size of each proxy and to increase the scalability of the caching storage.

1.5.3 Corporate Headquarters Buildings or Large Campuses

On large campuses or in corporate headquarter buildings, the size of the campus or number of personnel frequently lead to the creation of smaller local area networks interconnected by routers and a backbone LAN. Busy LAN servers combined with increasing use of Web server applications can result in congestion on the backbone LAN segments. This can be reduced by installing AFS client-enabled HTTP servers and general proxy caching on user LAN segments. The AFS clients can provide caching for corporate Web content, while the proxy server can provide caching and filtering for external Internet access.

Design considerations for the smaller LAN segments are similar to those for small branch offices above.

1.5.4 Backbone Internet Service Providers

Backbone Internet service providers typically provide co-location and/or peering services for other ISPs in addition to content hosting for large national or international corporations. In many cases they may provide *virtual ISP services* for other service providers such as content hosting ISPs. Backbone ISP customers are increasingly demanding both high availability and differentiated service levels and backbone ISPs are responding by enhancing their Internet infrastructures. Features demanded by backbone Internet service providers include:

- Load balancing for a variety of traffic (for example, mail and FTP in addition to Web traffic)

In addition, requests are made for traffic balancing management and high availability for authentication servers and management systems.

- Proxy caching

To minimize the effects of *hot potato* routing and Web traffic *surges*, backbone ISPs are typically installing highly scalable caches at major peering points and network interconnections points such as NAPs.

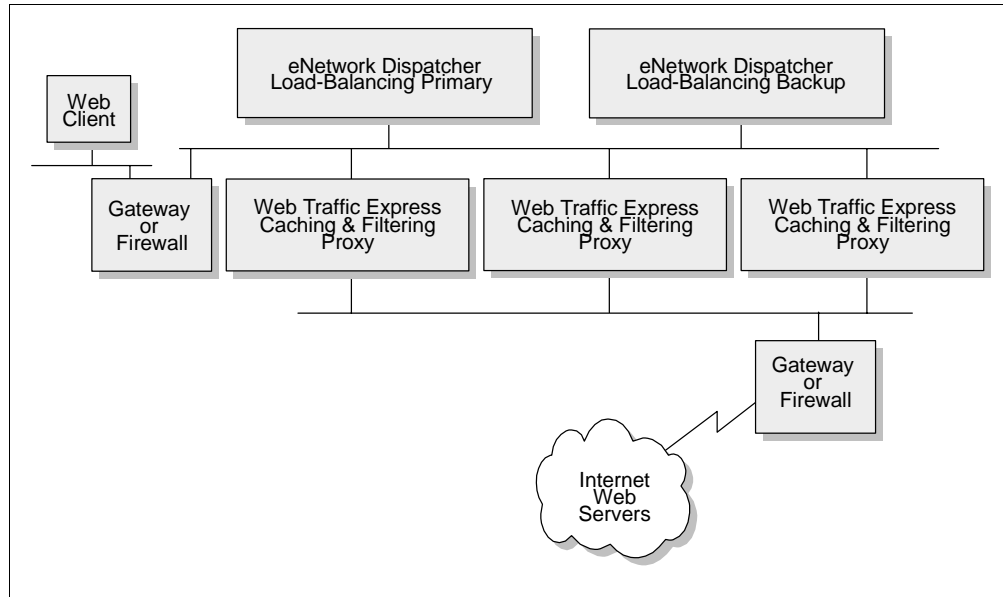


Figure 5. Hierarchical Caching

International ISPs in particular need caching to reduce the costs associated with trans-oceanic links. Such installations can make very effective use of the RCA feature of Distributed Web Traffic Express, a variation of the caching function that used together with load balancing and shared file storage can reduce redundancy of page storage.

Typical configurations for backbone ISPs would include load balancing for authentication gateways (such as authentication servers and subscriber management application servers) as well as for caching proxy servers. Because of the amount of traffic and the desire for high availability, such caching servers would likely use the RCA feature and would thus also be configured with AFS clients. Therefore there would also be an AFS server with the file content.

1.5.5 Access Internet Service Providers

Access Internet service providers need to provide both more consistent response time to their customers and to conserve their backbone network link and access charges. Thus caching at points of presence would address these needs. These configurations would be similar to those for backbone Internet service provider solutions.

Because access service providers target many of the small to medium-size business customers, there is also an opportunity for them to create revenue producing services, by deploying smaller caching devices on customer premises but configuring and managing them as an ISP service. For this scenario, the caching would look much as it does for corporate branch offices.

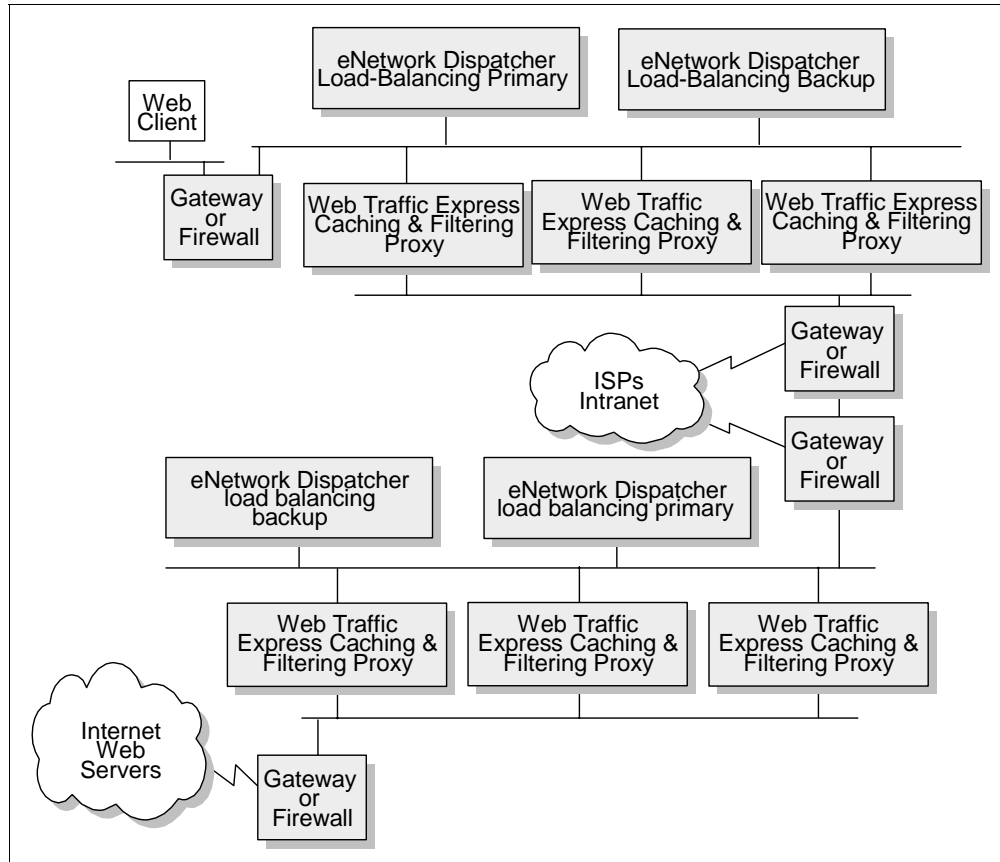


Figure 6. Access Internet Service Providers

1.5.6 Access ISPs with Subscriber Home Page Hosting

Access ISPs providing subscriber home page hosting typically do so on multi-homed servers, allowing subscribers to have their own domains. Such servers may have frequent changes, requiring the ISP to dedicate considerable time to administering the servers from a directory and backup perspective. For example, if the ISP is successful in recruiting new users, therefore having to expand the server capacity, it may involve reorganizing the various servers to spread the load and to allow room for individual sites to expand. The file management services provided by IBM WebSphere Performance Pack enable the ISP to reduce this maintenance workload, and to provide high availability replication for subscribers willing to pay for continuous availability. It also allows the ISP to scale up the site capacity with additional AFS clients, without the need to reorganize the underlying Web servers. Load balancing can ensure that despite the growth users get rapid response time. In addition, when combined with subscriber management routings, to intelligently assign DHCP IP addresses, it becomes possible to offer differentiated classes of services for customers paying a premium.

1.6 Other IBM WebSphere Offerings

This section describes the other IBM WebSphere products that can interact with IBM WebSphere Performance Pack in a Network Computing Framework environment: IBM WebSphere Application Server and IBM WebSphere Studio.

Notice that IBM WebSphere Performance Pack can be integrated with a number of other IBM e-business products, such as IBM eNetwork Firewall and Lotus Domino Go Webserver. In this redbook, there are several examples and scenarios where all these e-business products are used together to create a powerful and secure Web site.

For example:

- In 3.5, “Flexible-Client SOCKS” on page 117, we see how the Caching and Filtering component can be configured to optimize its interaction with IBM eNetwork Firewall.
- In 4.13, “Firewall High Availability Using the Load Balancing Component” on page 338, we show a successful and easy configuration where firewall high availability is obtained through the Load Balancing component of IBM WebSphere Performance Pack.
- In all the scenarios described in Part 2, “IBM WebSphere Performance Pack Scenarios” on page 347, we see how the content server functions implemented by Lotus Domino Go Webserver can be perfectly integrated with all the services provided by the IBM WebSphere Performance Pack components.

1.6.1 IBM WebSphere Application Server

IBM WebSphere Application Server lets you achieve your *write once, use everywhere* goal for servlet development. The product consists of a Java-based servlet engine that is independent of both your Web server and its underlying operating system. IBM WebSphere Application Server offers a choice of server plug-ins that are compatible with the most popular server application programming interfaces (APIs). The supported Web servers are:

- Lotus Domino Go Webserver
- Netscape Enterprise Server
- Netscape FastTrack Server
- Microsoft Internet Information Server
- Apache Server

In addition to the servlet engine and plug-ins, the Application Server provides:

- Implementation of the JavaSoft Java Servlet API, plus extensions of and additions to the API.
- Sample applications that demonstrate the basic classes and the extensions.
- The Application Server Manager, a graphical interface that makes it easy to set options for loading local and remote servlets, set initialization parameters, specify servlet aliases, create servlet chains and filters, monitor resources used by the Application Server, monitor loaded servlets and active servlet sessions, log servlet messages, and perform other servlet management tasks.

- A connection management feature that caches and reuses connections to your Java Database Connectivity (JDBC)-compliant databases. When a servlet needs a database connection, it can go to the pool of available connections. This eliminates the overhead required to open a new connection each time.
- Additional Java classes coded to the JavaBeans specification that allow programmers to access JDBC-compliant databases. These data access beans provide an enhanced function while hiding the complexity of dealing with relational databases. They can be used in a visual manner in an integrated development environment (IDE).
- Support for a new technology for dynamic page content called JavaServer Pages (JSP). JSP files can include any combination of inline Java, `<SERVLET>` tags, NCSA tags, and JavaBeans.
- CORBA support, an object request broker (ORB) and a set of services that are compliant with the Common Object Request Broker Architecture (CORBA).

1.6.2 IBM WebSphere Studio

IBM WebSphere Studio is a suite of tools that can be used by all the people involved in creating and maintaining Web sites. It allows your team to:

- Group your Web site files into projects and folders.
- Edit and update the files with your preferred tools.
- Easily create Java servlets and database queries using wizards.
- Publish all or part of the Web site on any of your WebSphere Application Servers.
- Maintain the files locally on individual workstations, or in a central location using a source control system.
- All work on the same projects. Content authors, graphic artists, programmers, and webmasters all have access to the files they need.

IBM WebSphere Studio is the tool complement of IBM WebSphere Application Server. This suite of tools makes it easier to design, develop and maintain dynamic, interactive Web sites.

The tools provided by IBM WebSphere Studio are:

- Servlet Generation Wizards for building Java servlets that can access JDBC databases and JavaBeans, as well as the associated HTML input and output forms
- The WebSphere Studio workbench, which provides a helpful Web site project organizer and a launch platform for preferred content tools, such as standard HTML editors and graphic design programs
- ScriptBuilder for text-based editing of HTML and script
- NetObjects Fusion Version 3 for easily building advanced-function Web sites
- VisualAge for Java Professional Edition Version 2, the advanced Java programming environment for building Web-enabled enterprise applications

Chapter 2. File Sharing Component

This chapter gets very specific about the File Sharing component of IBM WebSphere Performance Pack, also known as AFS. Notice that originally AFS was an acronym for Andrew File System, but now this term it is used as a single word.

The File Sharing component of IBM WebSphere Performance Pack provides file administration, scaleable content through AFS client caching, and non-disruptive file replication for various IBM WebSphere Performance Pack environments.

2.1 Features of the File Sharing Component

AFS is a *distributed file system* that allows users to share and access all of the files stored in a network of computers as easily as they access the files stored on their local machines. The file system is called distributed because files may reside on many different machines, but are available to users on every machine.

In fact, AFS stores files on a subset of the machines in a network, called *file server* or *File Sharing server* machines. File Server Machines provide file storage and delivery service, along with other specialized services, to the other subset of machines in the network, the client machines. These machines are called *File Sharing clients* because they make use of the servers' services while doing their own work. In a standard AFS configuration, clients provide computational power, access to the AFS files and other general purpose tools to the users seated at their consoles. There are generally many more client workstations than File Server Machines.

AFS servers run a number of *server processes*, so-called because each provides a distinct specialized service: one handles file requests, another tracks physical location of data, a third manages security, and so on. To avoid confusion, AFS documentation always refers to *server machines* and *server processes*, not simply to *servers*.

2.1.1 The Concept of AFS Cell

A *cell* is an administratively independent site running AFS. In terms of hardware, it consists of a collection of File Server Machines and client machines defined as *belonging to* the cell. Notice that a machine can only belong to one cell at a time. Users also belong to a cell in the sense of having an account in it, but unlike machines can belong to (have an account in) multiple cells.

To say that a cell is *administratively independent* means that its administrators determine many details of its configuration without having to consult administrators in other cells or a central authority. The administrator will make many decisions about setting up and maintaining your cell in the way that best serves its users. For example, a cell administrator determines how many machines of different types to run, where to put files in the local tree, how to associate volumes and directories, and how much space to allocate to each user.

2.1.2 Single File Name Space

Although your AFS cell is administratively independent, you will probably want to organize your local collection of files (your *file space* or *directory tree*) so that

users from other cells can also access this information. AFS allows cells to combine their local file spaces into a global file space, and does so in such a way that file access is transparent. Users do not need to know anything about a file's physical location in order to access the file. All they need to know is the pathname of the file, which includes the cell name. Thus every user at every machine sees the collection of files in the same way, meaning that AFS provides a uniform name space to its users.

2.1.3 AFS Volumes

AFS groups files into volumes, making it possible to distribute files across many machines and yet maintain a uniform name space. A *volume* is a unit of disk space that functions like a container for a set of related files, keeping them all together on one partition. In fact a volume is a collection of related files and directories and is part of the directory tree. This enables the directory tree to break the directory tree down into smaller units that can then be stored on file servers. Volumes can vary in size, but are by definition smaller than a partition, since partitions hold one or more volumes. We recommend a maximum size of 2 GB. You can read or write to AFS volumes that are larger than 2 GB, but you cannot perform typical AFS volume operation, such as dumping, restoring, moving or replicating the volume.

Volumes are important to system administrators and users for several reasons. Their small size makes them easy to move from one partition to another, or even between machines. The system administrator can maintain maximum efficiency by moving volumes to keep the load balanced evenly. In addition, volumes correspond to directories in the file space; most cells store the contents of each user home directory in a separate volume. Thus the complete contents of the directory move together when the volume moves, making it easy for AFS to keep track of where a file is at a certain time. Volume moves are recorded automatically, so users do not have to keep track of file locations.

2.1.4 Replication and Caching

AFS incorporates special features on server machines and client machines that help make it efficient and reliable:

- On server machines, AFS allows administrators to replicate frequently accessed but infrequently changed volumes, such as those containing binaries for popular programs. *Replication* means putting an identical ReadOnly copy (sometimes called a *replica*) of a volume on more than one File Server Machine. The crash of one File Server Machine housing the volume does not interrupt users' work, because the volume's contents are still available from other machines. Replication also means that one machine does not become overburdened with requests for files from a popular volume.

To benefit from AFS replication, at least two database servers are required. Three servers are considered even better. Database replication is obtained using a proprietary protocol named Ubik (see 2 on page 39). This is a different method from that used for data or volume replication.

- On client machines, AFS uses *caching* to improve efficiency. When a user on a client workstation requests a file, a process named Cache Manager, running on the client, sends a request for the data to the file server process running on the proper File Server Machine. The user does not need to know which machine this is; the Cache Manager determines file location automatically.

The Cache Manager receives the file from the file server and puts it into the cache, an area of the client machine's local disk or memory dedicated to temporary file storage.

Caching improves efficiency because the client does not need to send a request across the network every time the user wants the same file. Network traffic is minimized, and subsequent access to the file is especially fast because the file is stored locally.

2.1.5 Cached Data Validation

AFS has a way of ensuring that the cached file stays up-to-date, called a *callback*. When a user or process on an AFS client machine makes an update to a file, that file will be updated in the Cache Manager for an AFS client. When the file is changed, and the update is finished, the file is written back to the AFS server, in the appropriate ReadWrite volume. When a file server's copy of the file is updated, that server will notify each of the AFS clients that have a copy of that file. Those clients will be informed that a new copy of that file is available and they will need to retrieve that copy the next time a user or a process requests that specific file.

Notice that in order to copy modified data to each of the replicas, the system administrator must issue a particular command, `vos release`. The replication daemon, or `volserver`, handles this task, but it is not automatic. The administrator decides when to make this modification available and runs `vos release` to do this.

We see more details on this process in 2.7, "A Closer Look at AFS Cached Data Validation" on page 103.

2.1.6 Content Backup

Besides the volume that the user works with (the user's *home volume*), a snapshot of the user's directory is usually made every night. Thanks to the design of AFS, this snapshot is a volume, called a *backup volume*, that can also be attached to the file name space to allow easy retrieval of files mistakenly deleted. It is common for backup volumes to be created for all volumes in the system, not only user volumes, so the archive process can use this stable and consistent image when writing the data to tape. The administrator can set up the system, so users themselves can correct careless file deletions with no system intervention.

Notice that the creation of a backup volume every night is not automatic. The administrator sets this up if desired with a `bos cron` job. This is similar to a UNIX cron job in that it runs at a specified time, but under control of the BOS Server. For example, each server could have the following process running:

```
Instance backup, (type is cron) currently running normally.
Auxiliary status is: run next at Thu Nov  5 13:00:00 1998.
Process last started at Wed Nov  4 13:00:03 1998 (4 proc starts)
Last exit at Wed Nov  4 13:00:09 1998
Command 1 is '/usr/afs/bin/vos backupsys user. server1 -localauth'
Command 2 is '03:00'
```

You can obtain a similar output by entering the command:

```
bos status server1 -long
```

This process, which is run on server1, will create a backup copy of each volume whose name begins with user. on the server called server1 at 3:00 a.m. each morning.

2.1.7 AFS Security

Even in a cell where file sharing is especially frequent and widespread, it is not desirable that every user have equal access to every file. One way AFS provides adequate security is by requiring that servers and clients prove their identities to one another before they exchange information. This procedure, called *mutual authentication*, requires that both server and client demonstrate knowledge of shared secret information (such as a password) known only to the two of them. Mutual authentication guarantees that servers provide information only to authorized clients and that clients receive information only from legitimate servers.

AFS utilizes algorithms and other procedures based on *Kerberos*. This technology was originally developed by the Massachusetts Institute of Technology's Project Athena.

Users themselves control another aspect of AFS security, by determining who has access to the directories they own. Notice that every AFS directory has an ACL. For any directory a user owns, the user can build an *access control list* that grants or denies access to the contents of the directory. An access control list pairs specific users (or groups of users) with specific types of access rights.

As we see in 2.5.1, "Authenticating As admin" on page 86, there are seven separate access rights and up to twenty different users or groups of people may appear on an access control list.

2.2 Why Do I Need an Enterprise File System?

The File Sharing component of IBM WebSphere Performance Pack enables the easy sharing of information, and also builds a management infrastructure that supports ever-increasing file sharing demands without a proportional increase in costs. It's not just a way to share files, but it's a way for an organization to control its distributed file services.

From what we have seen in 2.1, "Features of the File Sharing Component" on page 17, the File Sharing component has many benefits, such as:

- High scalability because of architectural protocol efficiencies
- Ease of sharing via its single shared name space
- Reliable access to replicated data
- Single point of administration based on distributed services
- An efficient security model for access and group management
- Superior support for clients and desktops of different architectures

2.2.1 AFS and NFS

Network File System (NFS) from Sun Microsystems is another distributed file system option. NFS lacks many of the scalability and performance features available with AFS, and does not implement the common name space feature of

AFS. Other major differences between AFS and NFS are in the areas of performance, availability, management and security.

Interoperability between AFS and NFS is possible with Transarc's NFS/AFS Translator. This tool is particularly effective for NFS installations that are transitioning to AFS. Throughout a transition period, machines that have not been converted to AFS can access files stored in the AFS name space. The Translator also provides an AFS solution for those system types that are not directly supported with a full AFS port.

2.3 How the File Sharing Component Works

The name space concept starts with a typical desktop client computer. All AFS client machines run a Cache Manager process. The Cache Manager maintains information about the identities of the users logged into the machine, finds and requests data on their behalf, and keeps retrieved files on local disk. The local caching reduces network traffic and server load.

The central AFS file name space is key. While all the file and directory names in the name space are related to physical data on the servers, the names are visible in a single, enterprise-wide, and consistent directory structure. The servers perform all their work in concert to provide a seamless file name space visible by all desktops and all users. Any changes to the namespace, any new files or any changed data are publicized to clients immediately.

Unlike past systems, which made use of the `/etc/` file system on a client to map between a local directory name and a remote file system through a mounting operation, AFS does its mapping (file name to location) at the server. This process has the advantage of making the served file space location independent. Location independence means that a user does not need to know which File Server holds the file. The user only needs to know the path name of a file. Of course, the user does need to know the name of the AFS cell to which the file belongs. Use of the AFS cell name as the second part of the path name distinguishes between file name spaces of the local and non-local AFS cells.

The File Sharing component of IBM WebSphere Performance Pack has important applications in HTTP environments, as we see in Part 2, "IBM WebSphere Performance Pack Scenarios" on page 347.

Typically, an HTTP daemon sends Web browsers files available from the machine on which it executes, but the process does not care whether the files that it serves are stored on a local disk or on an AFS file server, since access to AFS files is indistinguishable from access to local files. Many HTTP servers will run with files on the local server, so when you see a path, it is mapping down into a file that is stored locally. From the HTTP server's perspective, there is nothing different from pulling something out of AFS. The Cache Manager and all the other elements that go into providing this unified namespace make it possible for HTTP to pull files out.

2.4 Installation and Configuration of the File Sharing Component

The File Sharing component of IBM WebSphere Performance Pack Version 1.0 is supported on three operating systems: IBM AIX and Sun Solaris support both the

File Sharing server and client; Microsoft Windows NT supports only File Sharing client.

In this section we show you how to install and configure the File Sharing component of IBM WebSphere Performance Pack on AIX and Windows NT. The steps to configure this component on Solaris are very similar to what we describe here for AIX. We describe to you our experience, step by step. We tried to experiment with several combinations using AIX and Windows NT. Typically, the first step is to install a machine that is an AFS server and contains also AFS client binaries. In the book *AFS Installation Guide* (shipped with the product, but not separately orderable), such a machine is sometimes called the *first AFS machine*. We chose to install and configure our first AFS machine as an AFS server and client simultaneously. This is possible on AIX, since AIX supports AFS server and client and even both the components together. For this reason, in this redbook, we often speak of the first AFS machine as a machine where the two components (AFS server and client) are simultaneously present.

Notice that on Windows NT only the AFS client component is supported, since the AFS server is not supported on this platform. For this reason, we installed and configured the first AFS machine on AIX. Then, we installed a stand-alone AFS client machine on AIX and another stand-alone AFS client machine on Windows NT.

2.4.1 Installation of the First AFS Machine on AIX

This section describes how to install the AFS server and client on the AIX platform.

Before starting our discussion, it would be good to speak about the hardware and software environment on which we performed our installation. The machine we used as our platform was a uniprocessor IBM RS/6000 43P having 196MB of RAM, 333 MHz of CPU, 8.6 GB of hard disk and one token-ring interface. We installed this machine with AIX Version 4.3. On this level of the AIX operating system, installing the File Sharing component of IBM WebSphere Performance Pack did not require any patches. The IP address of the machine where we performed this installation was 9.24.104.97, its host name was rs600030 and the domain name was itso.ral.ibm.com.

As we have already said, we chose to install the AFS server and the AFS client on the same machine. Then, we installed and configured the AFS client on a separate AIX machine (see 2.4.3, "Installation of a Stand-Alone AFS Client on AIX" on page 52), in order to see what differences occur in the configuration when the AFS client and server are installed on two different machines.

The AIX installation program for each component of IBM WebSphere Performance Pack makes use of Java InstallShield's setup class and so it requires that the Java Virtual Machine (JVM) Version 1.1 or later is installed on the system.

We did not need to install the JVM on our machine, because Java Development Kit (JDK) Version 1.1.2 is automatically installed with AIX Version 4.3. Notice that IBM WebSphere Performance Pack requires AIX Version 4.2.1 or later.

File Sharing Component and AIX 4.3.1

At the time this book went to print, the latest available version for the AIX operating system was 4.3.1. We tried to install AFS server and AFS client on this level of the operating system, but we found a problem configuring the file system, as we describe in step 1 on page 36. We did not have any other possibilities other than uninstalling AIX 4.3.1 and installing the previous Version 4.3, where we saw that the AFS component worked without any problems.

By entering the following command we noticed that the default installation of AIX 4.3 locates the JVM java executable file in the directory /usr/bin:

```
which java
```

Since JDK 1.1 or later is required on your system, you should also enter the command:

```
java -version
```

to discover the level of the JVM already installed on your machine. If needed, you should install the JDK 1.1. The CD-ROM of IBM WebSphere Performance Pack ships JDK 1.1.4 for AIX, which you can find in the directory JDK/AIX. A detailed description of how to install JDK 1.1 on AIX is found in the IBM redbook *Network Computing Framework Component Guide*, SG24-2119.

To prepare your AIX system for the installation, follow the steps listed below:

1. Insert the IBM WebSphere Performance Pack CD-ROM in the CD-ROM drive.
2. Log in as root.
3. From a command line, enter the following commands:

```
mkdir /CD-ROM  
mount -rv cdrfs /dev/cd0 /CD-ROM
```

4. Enter `cd /CD-ROM/AIX`.
5. To start the installation program, enter `java setup`.

After a while, you will get the Welcome window. After selecting the **Next** button, you are required to agree to all items of the software license. If you agree, check the box **Accept all terms of the license**, then click **Next**. You will see another window displaying the IBM WebSphere Performance Pack Version 1.0 readme file. We suggest you to take a look at that file as it contains interesting information about the product.

Click **Next**, and a dialog will be displayed where you can select the language only for the Load Balancing component. Notice that this option is offered to you even if you have not specified yet that you want to install that component.



Figure 7. Load Balancing Component Language Selection

After you click **Next**, you are prompted to enter the IBM WebSphere Performance Pack destination location. We accepted the default /public/WebSphere, and we had the setup program create such a directory, as shown in the next window:

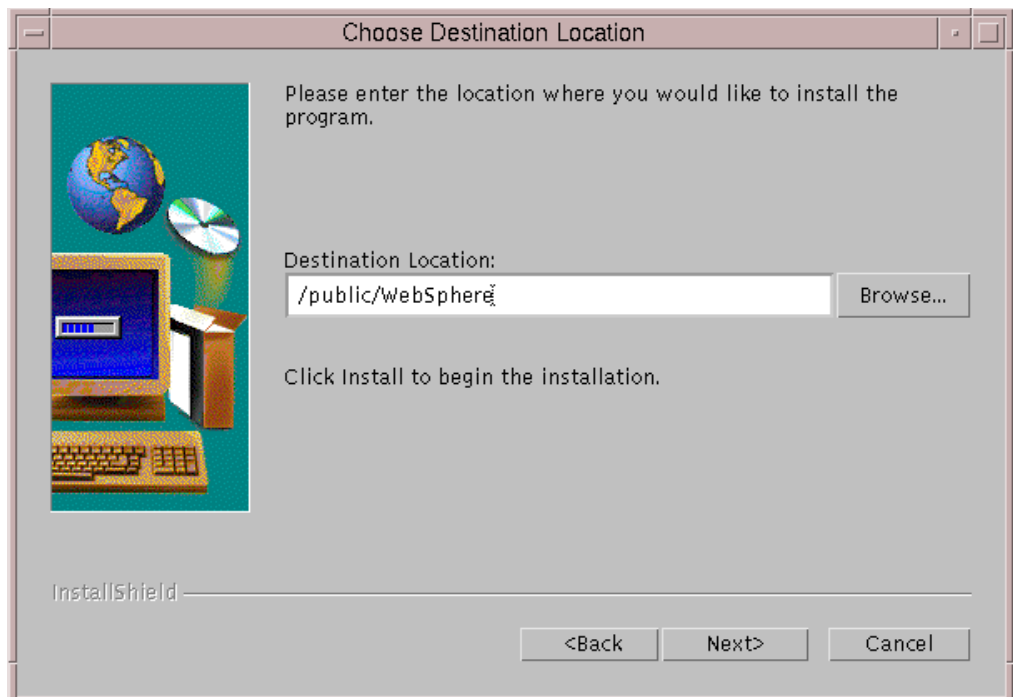


Figure 8. Choose Destination Location

In the above window you are asked to click **Install** to proceed, but such a button does not exist. Instead, select **Next** to continue.

Select now the IBM WebSphere Performance Pack components that you want to install. Note that the installation program allows you to install on AIX these combinations of components:

- Load Balancing (eNetwork Dispatcher) component and/or Caching and Filtering component and/or File Sharing client component
- File Sharing server component and/or File Sharing client component

Since this had to be our first AFS machine, we selected both **File Sharing Server** and **File Sharing Client**, as shown in the following screen:

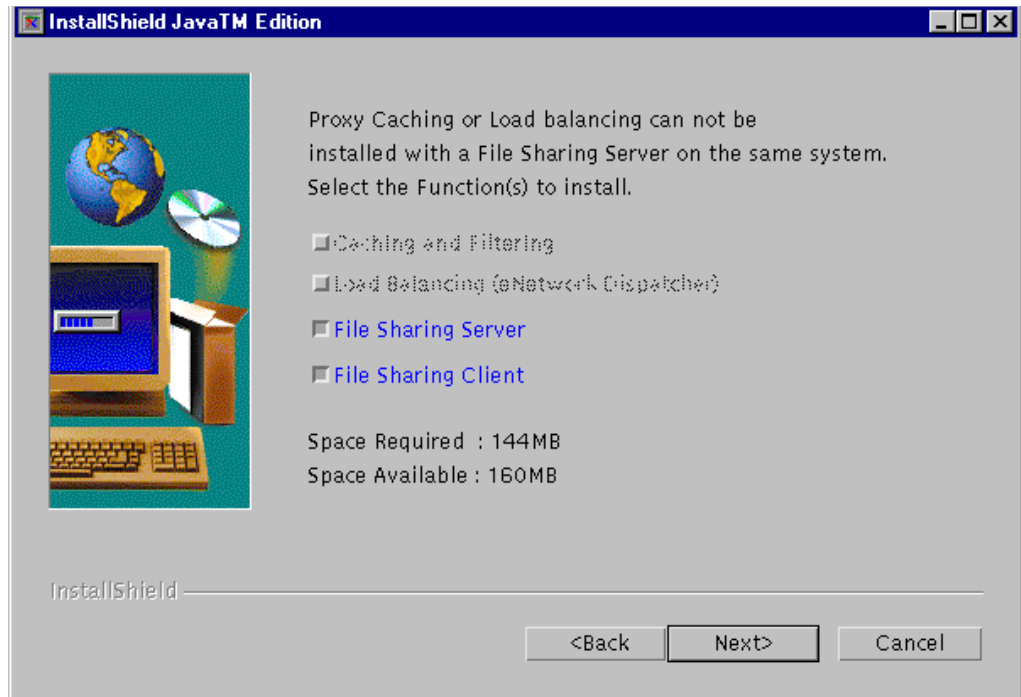


Figure 9. Select the File Sharing Server and File Sharing Client Components

Notice that as soon as we selected **File Sharing Server**, the two check boxes **Caching and Filtering** and **Load Balancing (eNetwork Dispatcher)** became immediately unavailable, since such components cannot be installed together with the File Sharing server component.

We had accepted to install the product in the /public/WebSphere directory (see Figure 8 on page 24).

As you can see in the above window, the disk space required by this installation is 144MB and the disk space available in the file system / on our system was 160MB. In this case, the installation could proceed. However, at the beginning of the installation, the situation was different. When we entered the command `df -k` we saw that the space available in the / file system wouldn't be enough for the installation. For this reason, we had to enlarge the / file system size, following the steps listed below:

1. From a command line, we entered:

```
smitty jfs
```

2. We selected **Change / Show Characteristics of a Journaled File System**.

3. We chose the file system `/`.
4. Since 144MB of free space are required (see Figure 9 on page 25), we set the size of the file system `/` to 425,984 512-byte blocks (see Figure 10 on page 26), which would be enough to install both the File Sharing server and File Sharing client.

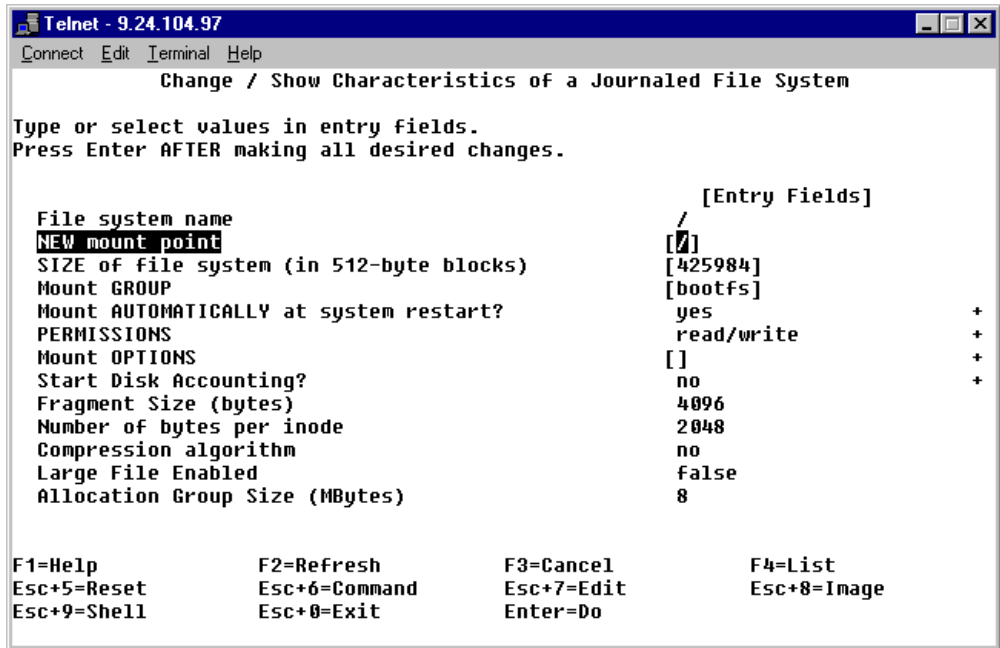


Figure 10. Enlarge the Size of the File System `/`

When the file system size is enough for your installation, you can choose the **Next** button in Figure 9 on page 25, and you will get a window that asks you if you want the installation program to replace other programs already installed on your system. We selected **No** in this case, since we had not installed any programs on our system yet.

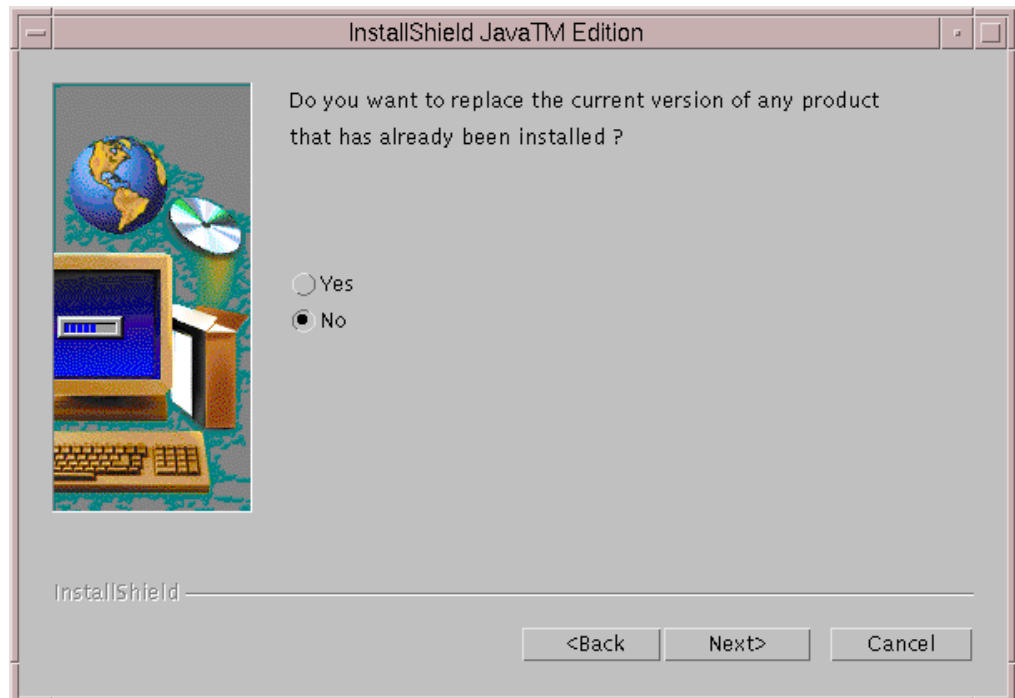


Figure 11. Choose to Not Replace Version

When we clicked **Next**, the following window was displayed:

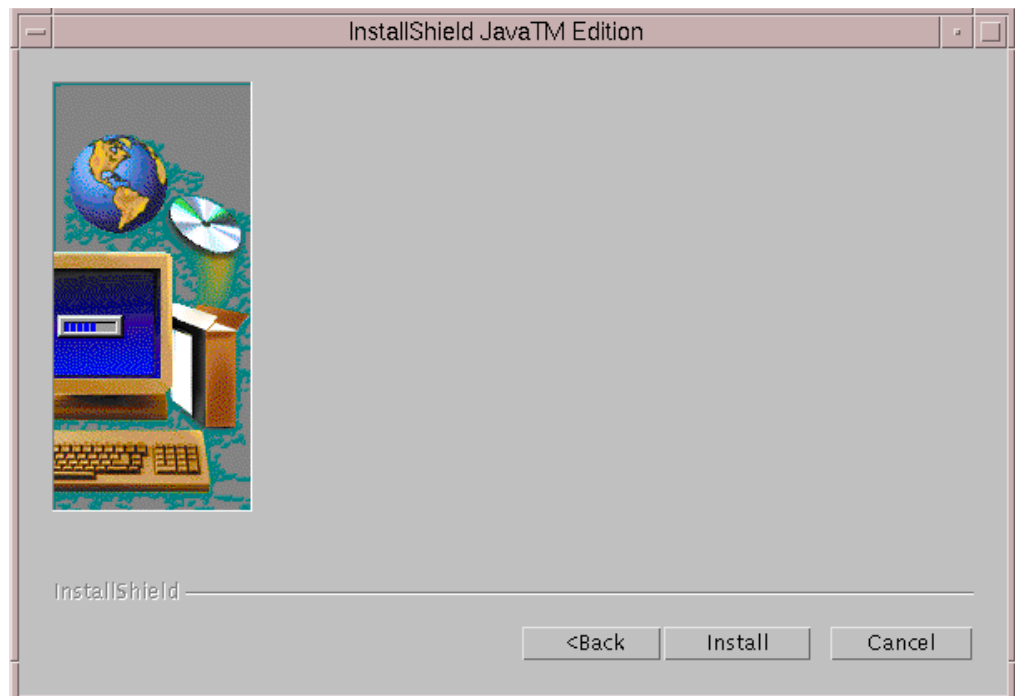


Figure 12. Start Installation Window

Click the **Install** button and after a while you will get informed that the installation is complete.

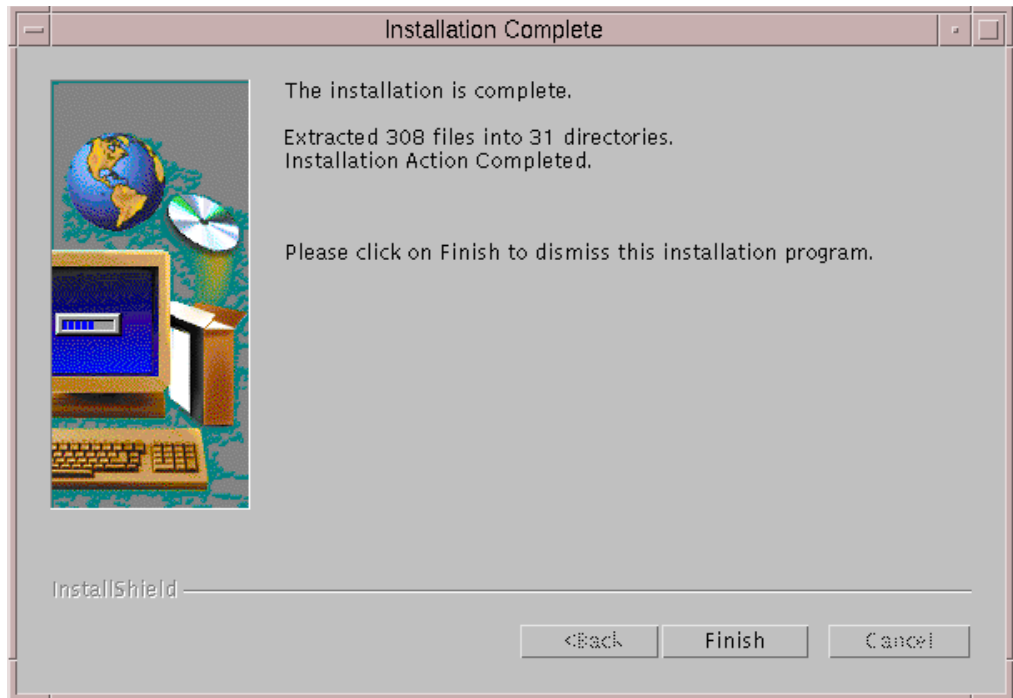


Figure 13. Installation Complete

Actually, even if the above window informs you that the installation is complete, there are still some other steps to do. First of all, after clicking **Finish**, you should verify that a set of files has been correctly copied in the directory `/public/WebSphere/AFS`. To do this, enter the command:

```
ls -l /public/WebSphere/AFS
```

You should see a window similar to the following:

```

Telnet - 9.24.104.97
Connect Edit Terminal Help
# ls -l /public/WebSphere/AFS
total 136
-rw-r--r--  1 root    system    10602 Sep 14 17:48 AFSinstall.class
-rw-r--r--  1 root    system     9680 Sep 14 17:48 afsConfigure.class
-rw-r--r--  1 root    system    4074 Sep 14 17:48 afsConfigureClient.class
-rw-r--r--  1 root    system    1024 Sep 14 17:48 afsConfigureServer.class
-rwxr-xr-x  1 root    system     427 Sep 14 19:08 afsConfigureServer.ksh
-rw-r--r--  1 root    system     411 Sep 14 18:30 afsConfigureServer.ksh.ori
-rw-r--r--  1 root    system    3082 Sep 14 17:48 afsCreateCache.class
-rw-r--r--  1 root    system     725 Sep 14 17:48 afsinstall.rc
-rw-r--r--  1 root    system     368 Sep 14 17:48 clean
-rw-r--r--  1 root    system     173 Sep 14 17:48 rc.afsd.large
-rw-r--r--  1 root    system     150 Sep 14 17:48 rc.afsd.medium
-rw-r--r--  1 root    system     243 Sep 14 17:48 rc.afsd.small
drwxr-x--x  3 root    system     512 Sep 14 17:48 usr
#

```

Figure 14. Verifying AFS Installation

Now you should change directory to the AFS directory by entering the following command:

```
cd /public/WebSphere/AFS
```

To complete the installation, you must type the `java` command against the `AFSinstall.class` file, by issuing:

```
java AFSinstall
```

After this step is complete, verify that the directories `afs` and `vice` have been created under the `/` directory. To do this, and see their contents at the same time, you can enter the following two commands:

```
ls -l /usr/afs
ls -l /usr/vice
```

2.4.2 Configuration of the First AFS Machine on AIX

In this section, we show you how to configure the first AFS machine on AIX. The platform where this configuration was performed is described in 2.4.1, “Installation of the First AFS Machine on AIX” on page 22.

2.4.2.1 Creation of an AFS Partition

First, we created an AFS partition on our AIX system. Notice that every AFS partition that you create must be associated with a directory, and all these directories *must* be called `/vicepx` where `x` is a lowercase letter. By convention the first directory is called `/vicepa`, the second `/vicepb`, and so on. AFS supports up to 256 vice partitions per server, named `/vicepa` to `/vicepz` and `/vicepaa` to `/vicepiv`.

Every File Server Machine *must* have at least one AFS partition. We created a directory called `/vicepa` by entering the command:

```
mkdir /vicepa
```

Then we created a journaled file system (JFS) by performing the following steps:

1. Log in as root.
2. Launch `smitty`.
3. Choose **System Storage Management (Physical and Logical Storage)**.
4. Choose **Logical Volume Manager**.
5. Choose **Logical Volumes**.
6. Choose **Add a Logical Volume** and then fill the fields as shown in the following figure:

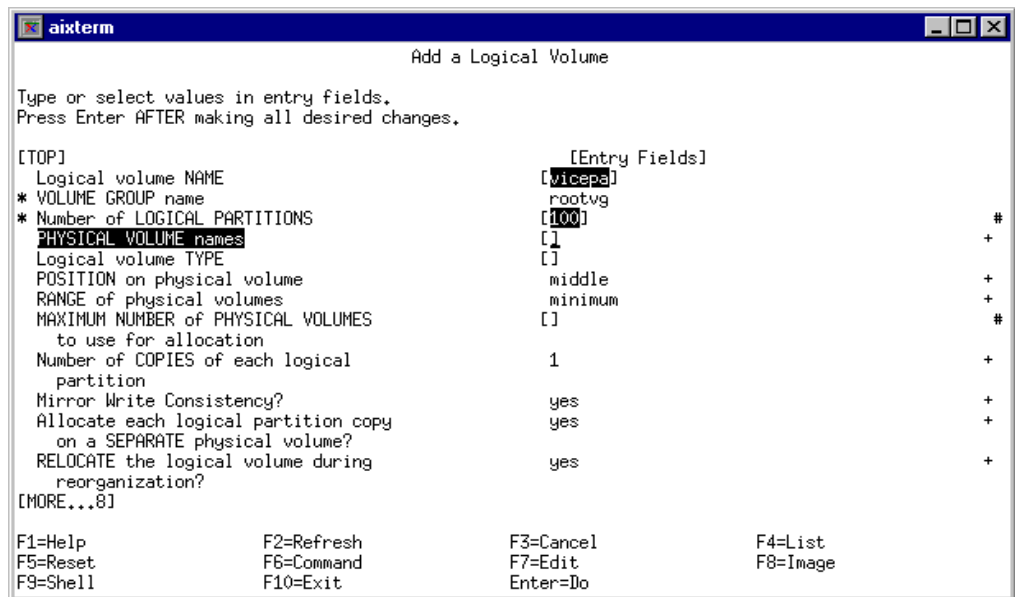


Figure 15. Add Logical Volume for AFS Partition

7. Press the Enter key to create the logical volume `vicepa`.
8. Press the F3 functional key until you return to the System Storage Management (Physical and Logical Storage) menu.
9. This time choose **File Systems**.
10. Choose the option **Add / Change / Show / Delete File Systems**.
11. Move the cursor to **Journaled File Systems**.
12. Choose **Add a Journaled File System on a Previously Defined Logical Volume**.
13. Choose **Add a Standard Journaled File System** and complete the fields as show in the following figure:

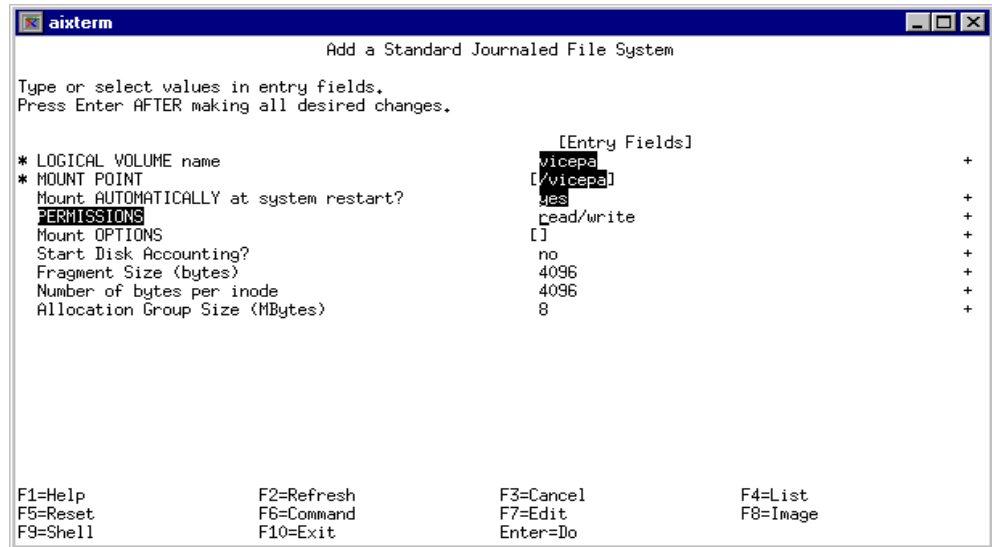


Figure 16. Create a Standard Journaled File System

Fragment Size

The above figure shows the file system created with a fragment size of 4096 bytes, that is 4KB. This is the smallest amount of allocated space. In other words, a file that is a single character will occupy 4KB.

AFS displays file system usage as if a 1KB fragment size is in use. Although it is acceptable to create the vice partition with a 4KB fragment size, just be aware that the numbers reported by AFS utilities such as `vos exam` and `vos listvol` will be incorrect.

14. Press the Enter key to create the file system.
15. Press the F3 functional key until you return to the file system's menu.
16. Choose **Mount a File System** and fill all the fields as shown in the following screen:

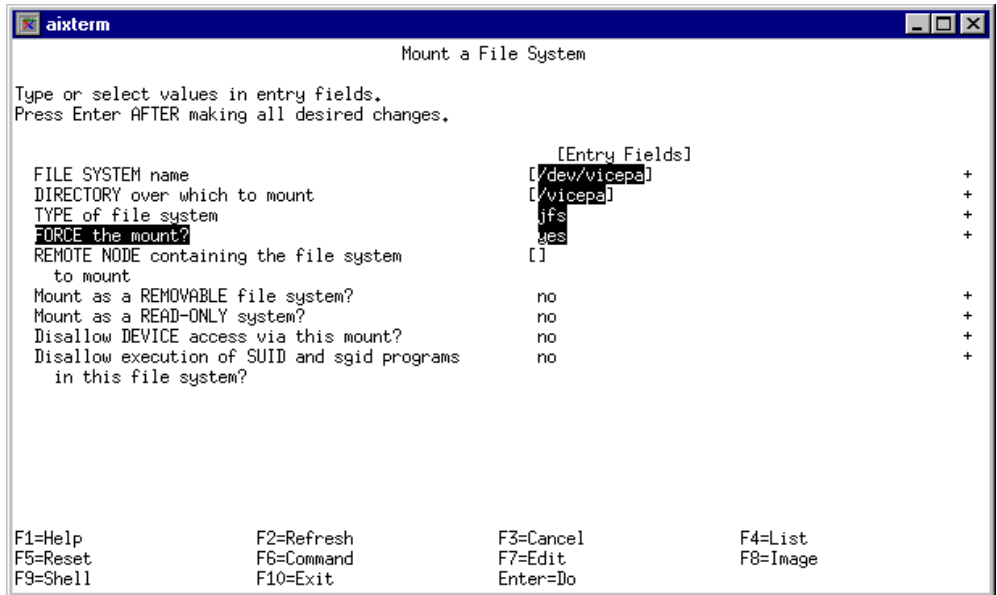


Figure 17. Mounting the Journaled File System JFS over the Directory /vicepa

17. Press the F3 functional key until you exit from smitty

Now, from an AIX command prompt, enter the `mount` command. This command lists all the mounted file systems. You should see the `/dev/vicepa` file system correctly mounted over the directory `/vicepa`.

Then, use a text editor such as `vi`, to edit the file `/etc/vfs`, which is a sort of configuration file for AIX file systems, and add the following line:

```
afs 4 none none
```

Then, save the file. This new line is very important when we create the AFS file system and mount it over the `/afs` directory. If such a line has not been added, the `mount` command will produce the following error message:

```
mount: 0506-309 /afs has a gfstype 4 that is not known
```

2.4.2.2 The fsck Program

Another important point that we want to emphasize is that you should never run the standard `fsck` program on an AFS File Server Machine. The `fsck` program is a very sophisticated file system integrity check that can usually recover problems with damaged file systems. This utility, if run on an AFS partition (or a partition housing an AFS volume), will discard the files that make up AFS volumes on that partition. Recall that AFS partitions are associated with the `/vicepx` directories, and they are not standard UNIX partitions. You should replace standard `fsck` with a modified `fsck` provided by Transarc, which properly checks both AFS and standard UNIX partitions.

For AIX systems, you do not replace `fsck` itself, but rather the `fsck` program helper distributed as `/sbin/helpers/v3fshelper`. In the book *AFS Installation Guide* (shipped with the product, but not separately orderable), we read that you should move the standard `fsck` program helper to a save file and then install the AFS-modified helper `/usr/afs/bin/v3fshelper` in the standard location. This operation can be accomplished through the following three commands:

```
cd /sbin/helpers
mv v3fshelper v3fshelper.noafs
cp /usr/afs/bin/v3fshelper v3fshelper
```

However, when we went to the /sbin/helpers directory and listed its contents, we found the following:

```
-r-xr-xr-x  1 bin      bin      168634 Sep 14 18:49 v3fshelper
-r-xr-xr-x  1 root    system   168634 Sep 14 18:49 v3fshelper.noafs
```

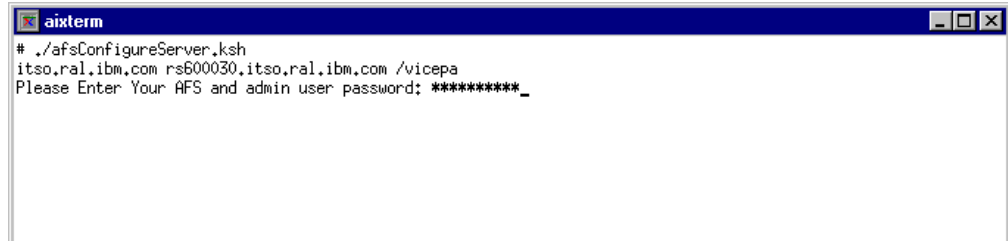
In other words, we saw that a file named v3fshelper.noafs had already been created, besides the file v3fshelper. The date and time when the two files v3fshelper and v3fshelper.noafs had been modified the last time were exactly the date and time when we had installed the AFS component of IBM WebSphere Performance Pack, so we understood that, although you should replace the fsck program helper for any AFS server typical installation, if you install AFS server as a component of IBM WebSphere Performance Pack, such a replacement is done automatically as part of the installation. We also noticed, as a confirmation, that the AFS-modified helper v3fshelper that we should have found under the /usr/afs/bin directory was not present on our system. For this reason, we did not have to replace the fsck program helper on our AIX system.

2.4.2.3 Configuration of the afsConfigureServer.ksh Script File

The next step was to edit the /public/WebSphere/AFS/afsConfigureServer.ksh file and customize the values of the following variables, according with your AFS server configuration:

1. The variable `CELLNAME` indicates the name of the cell to which the AFS server belongs. A cell is an independent and administrated site running AFS.
2. The variable `MACHINE_NAME` indicates the host name of the machine on which the AFS Server resides.
3. The variable `PARTITION_NAME` indicates the name of the AFS partition.

Figure 18 on page 34 shows the afsConfigureServer.ksh file after we finished editing it using the vi editor. As you can see, we set `CELLNAME` to `itso.ral.ibm.com`, `MACHINE_NAME` to `rs600030.itso.ral.ibm.com` and `PARTITION_NAME` to `/vicepa`.



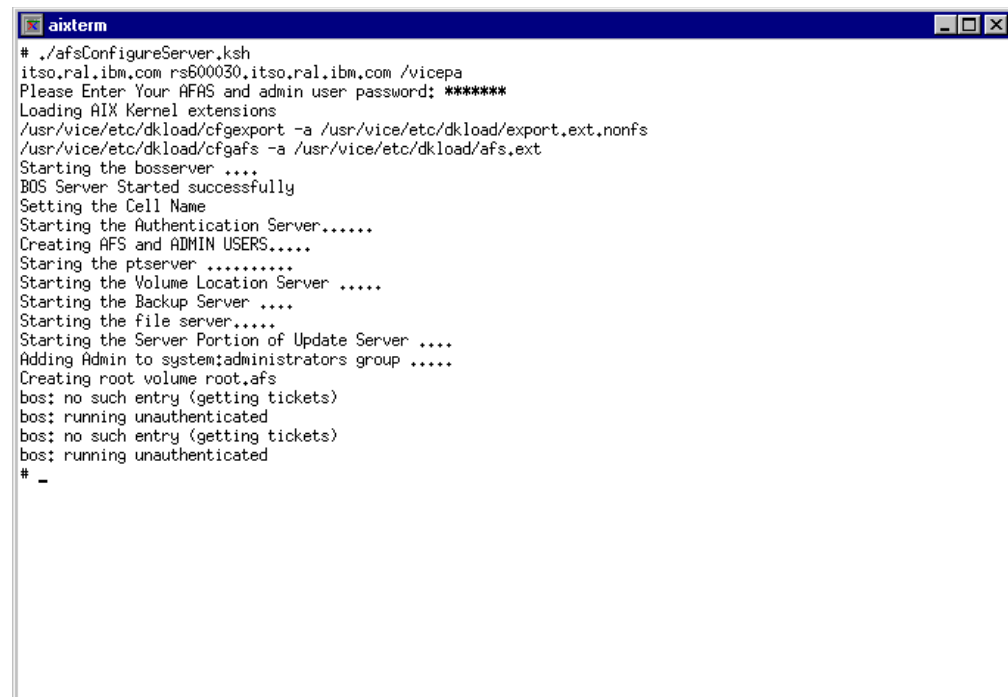
```
aiterm
# ./afsConfigureServer.ksh
itso.ral.ibm.com rs600030.itso.ral.ibm.com /vicepa
Please Enter Your AFS and admin user password: *****_
```

Figure 19. Prompt for AFS and admin User Password

The above figure shows the `afsConfigureServer.ksh` program while it is creating the admin user and the afs user. It's very important that *you don't forget this password* because you must use it in the rest of the configuration. We see its role later in this chapter.

Notice that the execution of this command can take a long time, because, as you can see from Figure 18 on page 34, this is a very complex executable file that starts all the AFS server processes and creates the admin account and the AFS cell.

At the end you will see this output:



```
aiterm
# ./afsConfigureServer.ksh
itso.ral.ibm.com rs600030.itso.ral.ibm.com /vicepa
Please Enter Your AFAS and admin user password: *****
Loading AIX Kernel extensions
/usr/vice/etc/dkload/cfgexport -a /usr/vice/etc/dkload/export.ext,nonfs
/usr/vice/etc/dkload/cfgafs -a /usr/vice/etc/dkload/afs.ext
Starting the bosservr ....
BOS Server Started successfully
Setting the Cell Name
Starting the Authentication Server.....
Creating AFS and ADMIN USERS.....
Starting the ptserver .....
Starting the Volume Location Server .....
Starting the Backup Server ....
Starting the file server.....
Starting the Server Portion of Update Server ....
Adding Admin to system:administrators group ....
Creating root volume root,afs
bos: no such entry (getting tickets)
bos: running unauthenticated
bos: no such entry (getting tickets)
bos: running unauthenticated
# _
```

Figure 20. Output of the `afsConfigureServer.ksh` Command

As you can see, there are many things that this program does. It launches several configuration commands, which remain hidden to the administrator. You can find the complete list of those commands in the book *AFS Installation Guide* (shipped with the product, but not separately orderable). You might want to enter each single command step by step, rather than running the `afsConfigureServer.ksh` script, which launches the commands on your behalf. If you chose to enter all the commands manually, you have the chance to understand all the steps of the

configuration and moreover you are able to realize if some of the configuration steps failed.

In our tests, we installed the AFS server and client several times on different machines and tried both the possibilities. In some cases we ran the `afsConfigureServer.ksh` script file, as shown in Figure 20 on page 35, and in other cases we issued the single commands individually. In this way, we were able to learn what each individual command does.

In the following list, we explain exactly what the `afsConfigureServer.ksh` script does. Our description is based on the experience we made with the list of the single commands. The output generated by `afsConfigureServer.ksh` is shown in Figure 20 on page 35, and we recommend that you see again that output while you read the description below.

1. First of all, read the following message:

```
Loading AIX Kernel extensions
```

Then `afsConfigureServer.ksh` executes the following two commands:

```
/usr/vice/etc/dkload/cfgexport -a /usr/vice/etc/dkload/export.ext.nonfs  
/usr/vice/etc/dkload/cfgafs -a /usr/vice/etc/dkload/afs.ext
```

Such commands execute the AIX kernel extension facility to load AFS into the kernel dynamically (kernel building is not possible on AIX). The first command specifically configures the machine to act only as an AFS machine and not as an NFS/AFS translator machine. We made this choice because our machine's kernel did not support NFS server functionality and we did not want our machine to be an NFS/AFS translator. Notice that all AFS server and client machines must run a kernel that incorporates AFS modifications. The preferred method for incorporating AFS into a machine's kernel is dynamic loading.

This operation must be run each time the machine reboots. As we will see later, it is recommended that you put these two commands in an initialization file that runs every time you reboot the machine. We show you how we created a file called `rc.afs` that had exactly this function.

The second command gave us several problems when we tried to configure the AFS server on AIX 4.3.1:

```
/usr/vice/etc/dkload/cfgafs -a /usr/vice/etc/dkload/afs.ext
```

This threw the following error message:

```
SYS_CFGKMODE: Invalid Argument
```

Since the kernel extension facility must be reloaded at each reboot into the kernel of every AFS machine, we considered this a serious problem and we decided to go ahead with AIX 4.3.0.

2. After loading AIX kernel extensions, the script file `afsConfigureServer.ksh` implicitly issues the command `bossserver`, which launches the Basic Overseer Server (BOS). The following message is displayed in the `aixterm` window:

```
Starting the BOS Server
```

The BOS Server ensures that all the server processes run correctly. The BOS Server can restart a process if this fails without contacting the administrator and helps the system administrator to manage configuration information.

3. Reading the message shown in Figure 20 on page 35, you can see that the `afsConfigureServer.ksh` program implicitly sets the cell name to `its0.ral.ibm.com`, according with the parameters we had set. The following message appeared on the aixterm window:

Setting the Cell Name

We noticed that at this point the `afsConfigureServer.ksh` script file also creates two files:

1. `/usr/afs/etc/ThisCell`, which defines the machine's cell membership and determines which cell's server processes the AFS command interpreters, generally running on client machines, should contact by default
2. `/usr/afs/etc/CellServDB`, which lists the cell's database machines.

Actually we found that the BOS Server also created `/usr/vice/etc/ThisCell` and `/usr/vice/etc/CellServDB` as symbolic links to the corresponding files in `/usr/afs/etc`. In fact, at the end of this process, if you change the current directory to `/usr/vice/etc` by issuing `cd /usr/vice/etc` and then enter the command `ls -la` you will see these entries for the files `ThisCell` and `CellServDB`:

```
lrwxrwxrwx 1 root system 23 Sep 14 19:45 CellServDB -> /usr/afs/etc/CellServDB
lrwxrwxrwx 1 root system 21 Sep 14 19:45 ThisCell -> /usr/afs/etc/ThisCell
```

Such symbolic links are necessary because the AFS command interpreters, which generally run on client machines, consult the `CellServDB` and `ThisCell` files in `/usr/vice/etc` for information about which cell's server processes they should contact. In an AFS server machine, these files currently reside only in `/usr/afs/etc` rather than `/usr/vice/etc`, and the links enable the command interpreters to retrieve the information they need. In an AFS stand-alone client machine, it is necessary that such files physically reside in `/usr/vice/etc`, as we see in 2.4.3, "Installation of a Stand-Alone AFS Client on AIX" on page 52.

4. Another important function of the `afsConfigureServer.ksh` script is to implicitly start the Authentication Server `kaserver`. We understood that this was happening when we read the following message, displayed in the aixterm window:

Starting the Authentication Server

The Authentication Server maintains the AFS Authentication Database, which records a password entry for each user in your cell. It also verifies the identity of users asking them for a password when they log into a cell. The `kaserver` grants the user a *token* that is used by any AFS clients to log in in the AFS system and prove their identity to AFS servers.

As we said in 2.1.7, "AFS Security" on page 20, the AFS Authentication Server is based on algorithms and procedures, based on Kerberos, a technology originally developed at the Massachusetts Institute of Technology.

5. The `afsConfigureServer.ksh` script file also creates in the AFS Authentication Database an entry `admin` for the user `admin`, the administrator of the AFS cell, and an entry `afs` for the user `afs`, which is for AFS server processes. This happens when the following message appears in the aixterm window where you launched the `afsConfigureServer.ksh` script file:

Creating AFS and ADMIN USERS

The user admin acquires administrative privileges in this process. Moreover it is also at this step that the configuration script places the server encryption key into `/usr/afs/etc/KeyFile`.

Notice that none will ever log in as afs, but the Authentication Server's Ticket Granting Service (TGS) module uses the password field to encrypt the server tickets that AFS clients must present to servers during mutual authentication.

6. The following output message indicates that the Protection Server process, `ptserver`, is starting:

```
Starting the ptserver
```

This process maintains the Protection Database, which stores AFS user names and user IDs, AFS group names and AFS group IDs and AFS group membership info. Using the command `/usr/afs/bin/pts` you can update the Protection Database, which is read by the `ptserver` when it starts. The command `/usr/afs/bin/fs` can create an access control list (ACL) for each directory in the cell. The Protection Server's main task is to help the file server determine whether a user has the correct rights to access a specified file.

7. The following output message indicates that the Volume Location Server process, `vlserver`, is starting:

```
Starting the Volume Location Server
```

This server maintains the Volume Location Database, where all information concerning the AFS volumes (such as volume names, physical locations and ID numbers) is stored. The Volume Location Database is aware of which File Server Machine in the cell houses the volume that contains the files that you are searching for. The `vlserver` process only runs on database servers. When a user tries to access an AFS directory, a process called Cache Manager interprets the path to learn which volume the file resides in. It contacts the Volume Location Server in order to learn which File Server Machine houses the volume. If information from the Volume Location Database is not available, the client can't access the requested directory.

8. The following output message indicates that the Backup Server process, `buserver`, is starting:

```
Starting the Backup Server
```

This process only runs on database servers. It maintains the Backup Database. The Backup Database records information about the volumes that are dumped to tape together, the schedules that are used, tape contents, etc.

9. The following output message indicates that the process `fs` is starting:

```
Starting the file server
```

This process *binds together* three server processes named File Server, Volume Server and Salvager:

1. `fileserv` - File Server process

It handles requests for file data at file/directory level.

2. `volserver` - Volume Server process

It handles requests at volume level, such as move, delete, create or replicate volumes.

3. `salvager` - Salvager process

This process attempts to repair disk corruption that can follow a crash.

A server machine gets different names determined by the processes it runs:

1. It is called a *File Server Machine* if it runs only the basic set of processes necessary for storing and delivering files. A File Server Machine also runs the BOS Server, which is a process that monitor's process status.
2. It is named *Database Server Machine* if it simultaneously runs all four AFS databases: Protection Database, Volume Location Database, Authentication Database and Backup Database. In your cell, to improve database availability, you can have multiple Database Server Machines, which are synchronized together using a proprietary protocol called Ubik.
3. It gets the name of *System Control Machine* if it stores and distributes system configuration information shared by all file systems in the cell. This information is stored in the /usr/afs/etc directory. Notice that files in the /usr/afs/etc directory must be identical on every server in your cell.
4. It is named *Binary Distribution Machine* if it stores and distributes binary files for the AFS commands to all the AFS servers running the same operating system.

In our scenarios (see Part 2, "IBM WebSphere Performance Pack Scenarios" on page 347), we have a single File Server Machine that is configured as a File Server Machine and Database Server Machine.

10. When the following output message is displayed, you know that the configuration process is starting the server portion of Update Server process, upserver:

```
Starting the Server Portion of Update Server
```

The server portion of Update Server is used to distribute the contents of /usr/afs/etc (from a System Control Machine) or /usr/afs/bin (from a Binary Distribution Machine) to other servers running the corresponding upclient process. In fact system performances would be inadequate if some machines were running different versions of a particular server process, such as, for example, Backup Server. To be sure that all server processes have the same version, the AFS installation on each server machine creates a server portion of Update Server, the upclient process. All the other machines of that type run the client portion of the Update Server. These two portions check frequently with each other to verify that the contents of their /usr/afs/etc and/or /usr/afs/bin directories are the same. If not, the client portion automatically retrieves the right version and installs it locally. For this reason, the system administrator does not have to remember to install new software individually on all the File Server Machines; it is the Update Server processes that do it automatically.

11. The following output message informs you that the user admin is going to be added to the system:administrators group:

```
Adding Admin to the system:administrators group
```

Members of this group can issue any `fs` or `pts` command. The group can also be placed on ACLs.

12. The following output message informs you that the `afsConfigureServer.ksh` script file is creating the `root.afs` root volume in the `/vicepa` partition:

```
Creating root volume root.afs
```

The `root.afs` volume will be mounted later over the `/afs` directory.

Note

Every time you execute AFS-related commands, you will see error messages like the following (as shown in Figure 20 on page 35):

```
bos: no such entry (getting tickets)
bos: running unauthenticated
```

This will happen until you authenticate yourself as admin in your cell. We show you how to log in as admin later in this section.

2.4.2.5 Verifying the Configuration

To verify that the configuration has completed successfully, follow this procedure. First, add `/usr/afs/bin` to your PATH system environment variable. To do this, edit the file `/etc/environment` (for example using the vi editor) and add `/usr/afs/bin` to the value of the PATH variable, as shown:

```
PATH=/usr/bin:/etc:/usr/sbin:/usr/ucb:/usr/bin/X11:/sbin:/usr/afs/bin
```

Then, log in again as root in order to have the system read the new PATH configuration. From now on, every time you log in as root, you will be able to execute commands in the directory `/usr/afs/bin` without having to specify the full path. Note that we found a little problem with the `/usr/afs/bin/fs` command, since for this particular command it was still necessary to specify the full path each time, although the directory `/usr/afs/bin` had been added to the value of the PATH variable. The reason for this was that in our system we had another executable file named `fs`, located in the directory `/usr/bin/X11`, which was part of the value of PATH as well.

Then, to display the status of the AFS Server processes, type the following and press the Enter key:

```
bos status rs600030 -long
```

You should see an output similar to the following screen:

```

aixterm
# ./bos status rs600030 -long
bos: no such entry (getting tickets)
bos: running unauthenticated
Bosserv reports inappropriate access on server directories
Instance kaserver, (type is simple) currently running normally.
  Process last started at Mon Sep 14 19:29:13 1998 (1 proc starts)
  Command 1 is '/usr/afs/bin/kaserver'

Instance ptserver, (type is simple) currently running normally.
  Process last started at Mon Sep 14 19:31:14 1998 (1 proc starts)
  Command 1 is '/usr/afs/bin/ptserver'

Instance vlserver, (type is simple) currently running normally.
  Process last started at Mon Sep 14 19:32:15 1998 (1 proc starts)
  Command 1 is '/usr/afs/bin/vlserver'

Instance buserver, (type is simple) currently running normally.
  Process last started at Mon Sep 14 19:34:15 1998 (1 proc starts)
  Command 1 is '/usr/afs/bin/buserver'

Instance fs, (type is fs) currently running normally.
  Auxiliary status is: file server running.
  Process last started at Mon Sep 14 19:36:15 1998 (2 proc starts)
  Command 1 is '/usr/afs/bin/fileserver'
  Command 2 is '/usr/afs/bin/volserver'
  Command 3 is '/usr/afs/bin/salvager'

Instance upserver, (type is simple) currently running normally.
  Process last started at Mon Sep 14 19:43:15 1998 (1 proc starts)
  Command 1 is '/usr/afs/bin/upserver -crypt /usr/afs/etc -clear /usr/afs/bin'

Instance runntp, (type is simple) currently running normally.
  Process last started at Mon Sep 14 19:43:16 1998 (1 proc starts)
  Command 1 is '/usr/afs/bin/runntp -localclock'

# -

```

Figure 21. Status of the AFS Server Processes

Now you should verify that an entry for the user admin has been successfully created in the AFS Authentication Database. To do this, enter:

```
kas examine admin
```

The output that you should get from this command is shown in the following screen:

```

aixterm
# kas examine admin
Password for root:
kas:examine: Auth, as root to AuthServer failed: user doesn't exist
Proceeding w/o authentication

User data for admin (ADMIN)
key (0):\316\235\364\351\337\313kd, last cpw: Mon Sep 14 19:31:13 1998
password will never expire.
An unlimited number of unsuccessful authentications is permitted.
entry never expires. Max ticket lifetime 25.00 hours.
last mod on Mon Sep 14 19:31:14 1998 by <none>
permit password reuse

# -

```

Figure 22. Checking if the User admin Has Been Successfully Created

Notice that the above command gives you information about the key and the password of the user admin.

To see in which group(s) admin belongs to, enter:

```
pts membership admin
```

You should see this response:

```
Groups admin (id: 1) is a member of:  
system:administrators
```

At this point, you can log on as admin by entering:

```
klog admin
```

You will be prompted with admin's password. Type it in and then press the Enter key.

2.4.2.6 Cache Manager Configuration

As we have already mentioned, our first AFS machine was both an AFS server and an AFS client. Every AFS client machine dedicates a portion of its local disk to a cache where it temporarily stores the data. Whenever an application program (for example, a text editor) running on a client machine requests data from an AFS space, this request passes through the Cache Manager. The Cache Manager is the portion of the client machine's kernel that translates file requests from a local application into a request to the server process, running on the File Server Machine that physically stores the file. When the Cache Manager gets the file for the first time, it stores the file in its cache and then passes it to the application. Notice that all this exchange of information is transparent to the user.

Every AFS client must have a copy of the files `ThisCell` and `CellServDB` in the `/usr/vice/etc` directory. Recall, the AFS server machines store these files in their `/usr/afs/etc` directory. The `ThisCell` file defines the machine's cell membership; this information is used by all the AFS programs that run on the client machine. The `CellServDB` file lists the Database Server Machines (whose number can be one or more) that the Cache Manager can contact. This includes database servers in the local cell, as well as foreign cells. If a cell's information is not in the file, the client cannot access data from that cell.

Notice that the Cache Manager consults `/usr/vice/etc/CellServDB` only once per reboot. As soon as the Cache Manager is initialized, the contents of `CellServDB` are copied into the kernel and, until the next reboot, the Cache Manager consults the list of Database Server Machines stored in the kernel.

In an AFS server machine, such files are automatically generated in the `/usr/afs/etc` directory when the `afsConfigureServer.ksh` script file is executed, and a symbolic link to these files is created in the `/usr/vice/etc` directory of the same machine (see step 3 on page 37). For this reason, if you are configuring your first AFS machine (which is both AFS server and AFS client), you will find symbolic links to these files already installed in the `/usr/vice/etc` directory after running the `afsConfigureServer.ksh` script file. While in a stand-alone AFS client machine it is necessary that actual copies of `ThisCell` and `CellServDB` are stored in the `/usr/vice/etc` directory, we saw that, in our first AFS machine, it was not necessary to replace the links with actual files.

As an example, we show you the files `/usr/afs/etc/ThisCell` and `/usr/afs/etc/CellServDB` that we found in our first AFS machine.

The following line was the content of the `/usr/vice/etc/ThisCell` file:

```
itso.ral.ibm.com
```

The contents of the `/usr/vice/etc/CellServDB` file were two lines:


```
>itso.ral.ibm.com      #Cell name
9.24.104.97           #rs600030.itso.ral.ibm.com
```

Notice that in the CellServDB file the first line must always begin with the > character, followed by the cell's Internet domain name. The domain name can be followed by a # sign and a comment that explains the name. Each subsequent line lists one Database Server Machine in the cell, specifying the Internet address of the server, followed by a # sign and the machine's host name. In this case, the # sign does not introduce a comment; the host name that follows is a required field. In fact the Cache Manager first attempts to contact the cell using the specified name, referring to the Internet address only if referring by name fails.

As we said, every AFS client must have a cache to store local copies of all the files brought over from File Server Machines. The Cache Manager can cache either on disk or in machine memory, but we chose to have a disk cache. For both types of caching, the afsd Cache Manager process consults the /usr/vice/etc/cacheinfo file that initializes the Cache Manager. Such a file must be created by the administrator. The following steps explain how to create the file.

First, we created two directories, /usr/vice/cache and /afs.

1. /usr/vice/cache will be the directory on the AFS client machine where by default the Cache Manager stores the files that it gets from the File Server Machine.
2. /afs will direct the Cache Manager to the cell's root.afs root volume, that you created previously through the afsConfigureServer.ksh script file, while you were configuring the AFS server (see 12 on page 39).

To create the above directories, we entered the following two commands:

```
mkdir /usr/vice/cache
mkdir /afs
```

Then we created the /usr/vice/etc/cacheinfo file. This file contains three fields, separated by colons. Such fields are described in the following list:

1. The first field specifies where to mount the AFS file system on the local disk. We made the standard choice, that is the directory /afs.
2. The second field defines the local disk directory that is used for caching. We chose /usr/vice/cache.
3. The third field defines the cache size as a number of kilobytes (1KB = 1024 bit). It's better to make it as large as possible, according to the disk space available. We chose to start with 100,000KB. It is important that you control the disk space available under the /usr file system before you set the cache size. Remember that you should specify 10% less (15% for AIX) than the space available to allow for overhead.

To create the cacheinfo file, enter:

```
echo "/afs:/usr/vice/cache:100000" > /usr/vice/etc/cacheinfo
```

Notice that in the book *IBM WebSphere Performance Pack for Multiplatforms - Getting Started* (shipped with the product but not separately orderable) you are

asked to specify also the partition name as a fourth field in the above cacheinfo file, but in the book *AFS Installation Guide* (shipped with the product, but not separately orderable), we found that such a fourth field does not exist. We saw that things worked very well without specifying the partition name as the fourth field. For this reason, we recommend that you create the cacheinfo file specifying only three fields, as we have just shown, since everything worked on the first try in our case.

Now, if you want to check the contents of the cacheinfo file, you can issue this command:

```
more /usr/vice/etc/cacheinfo
```

You will see the following output:

```
/afs:/usr/vice/cache:100000
```

2.4.2.7 Starting the Cache Manager

After this step, you should start the afsd daemon, which initializes the Cache Manager. To do this, you can use one of the three scripts provided by WebSphere Performance Pack:

1. rc.afsd.large
2. rc.afsd.medium
3. rc.afsd.small

Such scripts come by default after the installation, and are located in the directory `/usr/vice/etc/dkload`. Actually we found identical files also located in the directory `/public/WebSphere/AFS`.

We chose the script file named `/usr/vice/etc/dkload/rc.afsd.large`, whose contents are shown below:

```
# for systems with quite a few (5-10) simultaneous users, 32 MB RAM
# ~100 MB cache.
/usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -nosettime -verbose -volumes 128 $*
```

Figure 23. Contents of the File `/usr/vice/etc/dkload/rc.afsd.large`

The difference between these three scripts is that you can set up the way the afsd daemon starts; it can start with different parameters depending on the power of your AFS machine and the usage that you planned for it.

Notice, in fact, that in the first line of `rc.afsd.large`, which is commented out, it is specified that this file is indicated for systems with at least 32MB of RAM, 100MB of cache and a number of users accessing the AFS cell included between 5 and 10. The file `rc.afsd.large` is more indicated for large systems. We chose this file because we assumed that in our scenarios we would have at least ten users. Moreover our machine was very powerful, since it had 192MB of RAM and we had set up our cache partition in the cacheinfo file to 100MB.

We show you the contents of the two files `rc.afsd.medium` and `rc.afsd.small`, which are more appropriate for medium and small AFS systems respectively, in case you prefer to use one of them:

```
# for systems with several (2-6) users, 16 MB RAM, >40 MB cache
/usr/vice/etc/afsd -stat 2800 -dcache 800 -daemons 3 -volumes 70 $*
```

Figure 24. Contents of the File `/usr/vice/etc/dkload/rc.afsd.medium`

```
# for small systems, small cache. limited memory, light usage
# probably one or two users at most
# these settings are identical to the AFS 3.1b defaults
/usr/vice/etc/afsd -stat 300 -dcache 100 -daemons 2 -volumes 50 $*
```

Figure 25. Contents of the File `/usr/vice/etc/dkload/rc.afsd.small`

We launched the Cache Manager through the file `rc.afsd.large` by entering the following command:

```
/usr/vice/etc/dkload/rc.afsd.large
```

As we show you later (see Figure 26 on page 50), we recall `rc.afsd.large` from within the boot file that we write for our system, so that it is automatically launched every time the machine reboots.

If you have set to have a disk cache, rather than a memory cache, starting up the `afsd` daemon for the first time can take up to ten minutes, because the Cache Manager has to create all the structures needed for caching. Starting `afsd` at future reboots will not take this long, since such structures already exist.

We noticed that in the book *AFS Installation Guide* (shipped with the product, but not separately orderable), you are suggested at this point to authenticate yourself as `admin`. We did not consider this operation necessary at this time, since all the server processes were still running unauthenticated. In other words, until the authentication checking is set to on, both `root` and `admin` can launch all the AFS commands that require administration authority. After the authentication checking is turned on, only `admin` has administration authority and can launch AFS commands that require this authority level.

To make sure that Cache Manager can update all information about the AFS volumes such as `/vicepa`, you can use the following command:

```
/usr/afs/bin/fs checkvolumes
```

This is the output you should see:

```
All volumeID/name mappings checked.
```

Do not attempt to access `/afs` (by issuing commands such as `cd` or `ls`), until you complete all the previous steps, or you will cause error messages to be generated. In fact it is only after the above steps have been completed that the Cache Manager starts and knows how to access the `/afs` directory.

2.4.2.8 The `afsConfigureClient` Java Class

We want to mention here that we had a problem when we tried to go on with the standard configuration of the File Sharing client component of our first AFS machine. The guide *IBM WebSphere Performance Pack for Multiplatforms -*

Getting Started (shipped with the product but not separately orderable) suggests that at this point you should launch the Java class `afsConfigureClient`, which is located in the directory `/public/WebSphere/AFS`. Such a class takes `MACHINE_NAME` and `PARTITION_NAME` as parameters, so the exact sequence of commands would be:

```
cd /public/WebSphere/AFS
java afsConfigureClient MACHINE_NAME PARTITION_NAME
```

The last command would automatically configure the AFS client on your machine. However, as soon as we tried to launch the `afsConfigureClient` Java class, a Java exception was thrown and we couldn't go on with the AFS client configuration. If a similar problem happens to you, a solution is to follow the steps described in the book *AFS Installation Guide* (shipped with the product, but not separately orderable). That is what we did, and everything worked fine on our platform. For this reason, we show you the sequence of the operations we performed and you can easily follow our successful experience as an example.

2.4.2.9 Setting an ACL on /afs

First we show you how we set an access control list (ACL) on `/afs`. To grant `system:anyuser` rights to read and look up the `/afs` directory, enter the following command:

```
/usr/afs/bin/fs setacl /afs system:anyuser rl
```

If you do not grant such rights to all AFS users, they may not be able to reach their home directory, which will be mounted under the `/afs` directory.

2.4.2.10 Working with a New Volume

After this step, we show you how to create a new volume, attach it to the AFS directory tree and set its ACL. These three operations are required for every volume that you want to add in your cell.

The volume `root.afs` was automatically created in the `/vicepa` partition through the `afsConfigureServer.ksh` script file (see 12 on page 39) and we have also explained how this volume was attached to the `/afs` directory (see 2 on page 43). We show you now how we manually created the volume `root.cell` in the same `/vicepa` partition, how we attached it to the directory tree and how we set its ACL.

Here is how we created the `root.cell` volume:

```
vos create rs600030 /vicepa root.cell -cell itso.ral.ibm.com
```

Notice that `-cell itso.ral.ibm.com` is not required on the `vos create` command, provided the client is a member of `itso.ral.ibm.com` (or `/usr/vice/etc/ThisCell` contains `itso.ral.ibm.com`).

We attached `root.cell` to the directory tree by creating a mount point. The mount point was `/afs/itso.ral.ibm.com`. The command we issued to create the mount point was:

```
/usr/afs/bin/fs mkmount /afs/itso.ral.ibm.com root.cell
```

We created this to be a *regular mount point*. This mount point tells the Cache Manager to access only the ReadWrite version of `root.cell`. By convention a ReadWrite point has a period in front of the directory name. For this reason, we entered:

```
/usr/afs/bin/fs mkmount /afs/.itso.ral.ibm.com root.cell -rw
```

Then we defined replication sites for root.afs and root.cell through the following two commands:

```
vos addsite rs600030 /vicepa root.afs
vos addsite rs600030 /vicepa root.cell
```

In this way, we made sure that every user would be able to access the ReadOnly version of our root.cell volume.

We used the fs commands to verify that root.afs and root.cell existed and were available. To check the status of root.afs we entered:

```
/usr/afs/bin/fs examine /afs
```

And this was the output that we got:

```
Volume status for vid = 536870922 named root.afs.readonly
Current disk quota is 5000
Current blocks used are 4
The partition has 791832 blocks available out of 819200
```

Then we entered the following command to check the status of root.cell:

```
/usr/afs/bin/fs examine /afs/itso.ral.ibm.com
```

This was the response we received:

```
Volume status for vid = 536870919 named root.cell.readonly
Current disk quota is 5000
Current blocks used are 6
The partition has 791832 blocks available out of 819200
```

Then we set an ACL for the root.cell volume giving read and lookup rights to the system:anyuser group:

```
/usr/afs/bin/fs setacl /afs/.itso.ral.ibm.com system:anyuser rl
```

When you issue this command, you must specify .itso.ral.ibm.com, so you access the ReadWrite volume. If we entered /afs/itso.ral.ibm.com rather than /afs/.itso.ral.ibm.com, we would have received an error message:

```
You cannot change a backup or readonly volume
```

To make sure that root.cell and root.afs exist now, you can access the directories /afs and /afs/itso.ral.ibm.com using commands such as cd or ls.

The changes we made were to the ReadWrite volume. To propagate these changes to the ReadOnly volume, we entered the following two commands to release the ReadOnly replicas of root.cell and root.afs, giving ReadOnly access to all AFS users:

```
vos release root.afs
vos release root.cell
```

Now, to verify that the local Cache Manager sees the ReadOnly versions of root.afs and root.cell, issue the following command:

```
/usr/afs/bin/fs checkvolumes
```

You should see this output:

```
All volumeID/name mappings checked.
```

To check the status of /afs after the release of the volumes, enter this command again:

```
/usr/afs/bin/fs examine /afs
```

You should see the following output:

```
Volume status for vid = 536870922 named root.afs.readonly
Current disk quota is 5000
Current blocks used are 4
The partition has 793280 blocks available out of 819200
```

Then you should also enter the following command to check the status of /afs/itso.ral.ibm.com after the release of the commands. In this case the output should be:

```
/usr/afs/bin/fs examine /afs/itso.ral.ibm.com
```

```
Volume status for vid = 536870919 named root.cell.readonly
Current disk quota is 5000
Current blocks used are 5
The partition has 793280 blocks available out of 819200
```

We want to remind you that you *must* give at least lookup rights to any user under the path /afs/itso.ral.ibm.com because all users' directories that you will create will be located under this path. If a user does not have lookup rights, such a user will not be permitted to go into its home directories.

2.4.2.11 Turning on Authorization Checking

The following command is used to turn on the authorization checking, so that server processes on this machine will only perform privileged actions for authorized users:

```
bos setauth rs600030 on -cell itso.ral.ibm.com
```

From this moment on, you must have administration authority (*tokens for admin*) to perform administration commands and actions. If you want to add an AFS user, for example, you must authenticate as admin by issuing the command:

```
klog admin
```

You will be prompted for admin's password. Type it and press the Enter key. After that, you will be able to define a new AFS user. We describe how this operation is performed in 2.5, "Creation of an AFS User" on page 85.

2.4.2.12 AFS Server Processes

At this point, we can summarize the AFS server processes, running on AFS server machines:

- AFS Database Server machines run the following server processes:
 1. The Volume Location Server process, vlserver, which maintains the Volume Location Database
 2. The Protection Server process, ptserver, which maintains the Protection Database
 3. The Authentication Server process, kaserver, which maintains the Authentication Database
 4. The Backup Server process, buserver, which maintains the Backup Database.
- AFS File Server Machines run an fs instance, which is composed of three processes:
 1. The File Server process, fileserver, which handles requests at file directory level
 2. The Volume Server process, volserver, which handles requests at volume level
 3. The Salvager process, salvager, that repairs corruption

Any of these servers can run the server portion upserver and/or the client portion upclient of the Update Server process.

The first AFS machine is required to be both a database and File Server Machine and will run all of the above processes.

2.4.2.13 Restarting the Server Processes

To finish your configuration you should now restart all the server processes, including the BOS Server, to make sure that they establish new connections with one another, obeying the new authorization checking requirements. To do this, you should issue the `bos restart` command, with the flag `-bosserver`. That forces the BOS Server to restart as well. Here is the command we issued on our platform:

```
bos restart rs600030 -bosserver -cell itso.ral.ibm.com
```

2.4.2.14 Writing the Boot File for the First AFS Machine

The last step is to write a boot file for AFS. This is an initialization file that is executed every time your machine reboots. We called it `/etc/rc.afs` and we included in it all the AFS configuration commands necessary to:

1. Load the AIX kernel extension
2. Start the BOS Server
3. Start the Cache Manager
4. Turn on authorization checking

Note that you *must* respect the order we have just indicated when you will write your own boot file. We show you here the `/etc/rc.afs` file that we installed on our machine. The above list explains what this file exactly does.

```

#!/bin/ksh
echo 'loading AFS ... ' > /dev/console
cd /usr/vice/etc/dkload
./cfgexport -a export.ext.nonfs
./cfgafs -a afs.ext
/usr/afs/bin/bosserver &
/usr/vice/etc/dkload/rc.afsd.large
/usr/afs/bin/bos setauth rs600030 on -cell itso.ral.ibm.com

```

Figure 26. Contents of the File /etc/rc.afs in Our AFS Client and Server AIX Machine

This script file is specific for a first AFS machine, since it includes commands that are related to both the AFS server and client components. If you chose to have a stand-alone AFS server machine, you should remove the following line from the initialization file:

```
/usr/vice/etc/dkload/rc.afsd.large
```

In fact, such a line invokes the rc.afsd.large file, which starts the Cache Manager, a specific component of the AFS client. We show you in 2.4.3, “Installation of a Stand-Alone AFS Client on AIX” on page 52 how to create a similar file on a stand-alone AFS client.

In the above ksh script, the `bos setauth` command is not always required. It is only required during the initial install because we originally start the BOS Server in noauth mode. If the server is in noauth mode, the file `/usr/afs/local/NoAuth` exists. If this file is not present, the server is functioning normally with authentication checking turned on. So, although it does not hurt to include the `bos setauth` command, it is not really required.

Next, we edited the file `/etc/inittab` to make sure that rc.afs would be started by AIX at each reboot. We added the following entry to the `/etc/inittab` file:

```
rcafs:2:wait:/etc/rc.afs >/dev/console 2>&1 # Start AFS
```

Do not forget to make `/etc/rc.afs` executable. This can be done by entering the command:

```
chmod 744 /etc/rc.afs
```

2.4.2.15 After Rebooting the First AFS Machine

At this point, you can reboot your machine. When the boot is complete you should check that all the AFS processes started correctly. To do this, we entered the command:

```
bos status -server rs600030
```

The following lines in the output confirmed that all the server processes were running normally:


```
Instance kaserver, currently running normally.
Instance ptserver, currently running normally.
Instance vlserver, currently running normally.
Instance buserver, currently running normally.
Instance fs, currently running normally.
Auxiliary status is: file server running.
Instance upserver, currently running normally.
Instance runntp, currently running normally.
```

We also entered the command to see exactly which AFS processes were effectively running:

```
ps -ef | more
```

We show you here the output that we got on our system after launching this command. Since we had installed and configured our machine to be the first AFS machine, which means AFS server and client in the same time, the `ps -ef` command showed some processes related to the AFS server component and others related to the AFS client component.

The following figure shows only the AFS server processes that should be displayed after entering the `ps -ef` command:

```
root 9292      1  0  Sep 21   -  0:00 /usr/afs/bin/bossserver
root 10068    9292  0  Sep 21   -  0:00 /usr/afs/bin/volserver
root 14706    9292  0  Sep 21   -  0:03 /usr/afs/bin/kaserver
root 14964    9292  0  Sep 21   -  0:03 /usr/afs/bin/ptserver
root 15222    9292  0  Sep 21   -  0:03 /usr/afs/bin/vlserver
root 15480    9292  0  Sep 21   -  0:02 /usr/afs/bin/buserver
root 15740    9292  0  Sep 21   -  0:02 /usr/afs/bin/fileserver
root 16254    9292  0  Sep 21   -  0:00 /usr/afs/bin/runntp -localclock
root 15996    9292  0  Sep 21   -  0:00 /usr/afs/bin/upserver -crypt /usr/afs/etc -clear /usr/afs/bin
```

Figure 27. AFS Server Processes Displayed by the `ps -ef` Command

The following figure shows only the AFS client processes that should be displayed after entering the `ps -ef` command. These processes are generated by the `afsd` daemon (see Figure 23 on page 44), so they demonstrate that the Cache Manager started correctly:

```

root 9038      1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 10322    1  0  Sep 28   -  0:09 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 10580    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 10836    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11094    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11352    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11610    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11868    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12126    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12384    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12642    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12900    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13158    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13416    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13674    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13932    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 14190    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 14448    1  0  Sep 28   -  0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128

```

Figure 28. AFS Client Processes on the First AFS Machine

You should also verify that the two AFS volumes `root.afs` and `root.cell` have been correctly attached. To do this, it is enough to access the directory `/afs` through the command:

```
cd /afs
```

Then issuing `ls -l` should list the two directories `.itso.ral.ibm.com` and `itso.ral.ibm.com`.

2.4.3 Installation of a Stand-Alone AFS Client on AIX

In 2.4.1, "Installation of the First AFS Machine on AIX" on page 22, we have shown how to install the first AFS machine and we have also described in 2.4.2, "Configuration of the First AFS Machine on AIX" on page 29, how to configure the AFS server and client in the same AIX machine. In this section we show you the steps we followed to install a stand-alone AFS client machine on AIX. A stand-alone AFS client is a machine where the AFS server has not been installed. On Windows NT, every AFS client is stand-alone, since the AFS server is not available on this platform yet (see 2.4.5, "Installation of a Stand-Alone AFS Client on Windows NT" on page 58).

In this experience we used an IBM RS/6000 7012-370 having 132MB of RAM, 62.5 MHz of CPU, 2.0 GB of hard disk and one token-ring interface. As we mentioned in 2.4.2, "Configuration of the First AFS Machine on AIX" on page 29, we were not able to perform the configuration of the AFS component of IBM WebSphere Performance Pack on AIX 4.3.1 (see step 1 on page 36). We installed our stand-alone client machine with AIX Version 4.3.0, and it was not necessary to install any patches on this platform.

We give you now other information about our environment that will be useful in the rest of this section. The host name of our machine was `rs600012` and its IP address was `9.24.104.124`. The domain name was `itso.ral.ibm.com`.

In order to install a stand-alone AFS client, you should follow the same steps described in 2.4.1, "Installation of the First AFS Machine on AIX" on page 22, the only difference being that you must only check the **File Sharing Client** check

box, when you are prompted to select the IBM WebSphere Performance Pack components you want to install, as shown in the following screen:

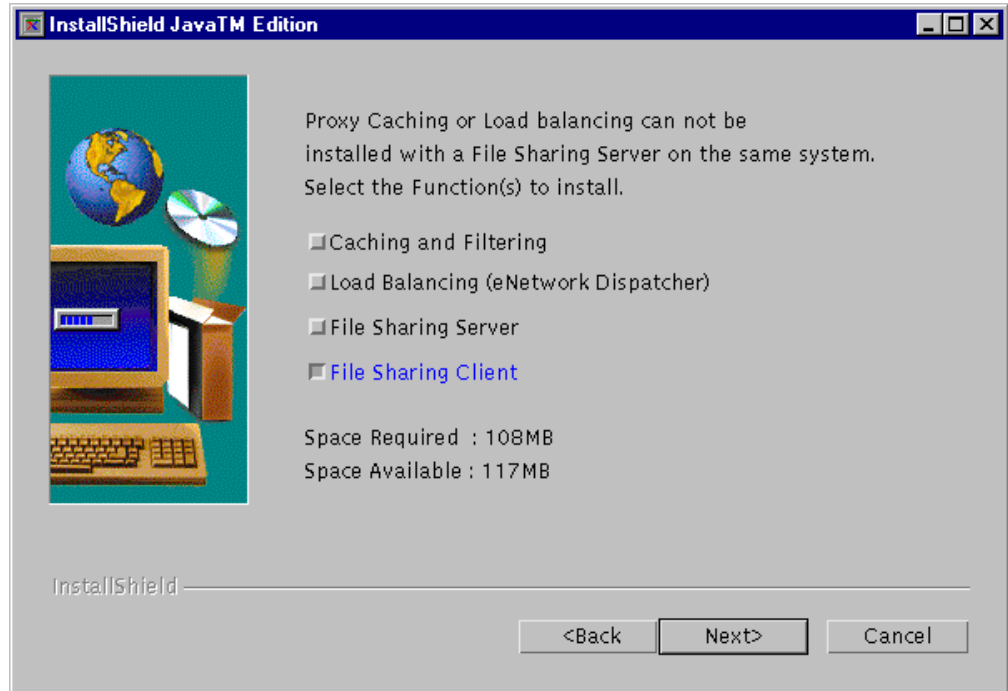


Figure 29. Installing a Stand-Alone AFS Client on AIX

As you can see from the above figure, the File Sharing client component on AIX is compatible with all the other components of IBM WebSphere Performance Pack.

2.4.4 Configuration of a Stand-Alone AFS Client on AIX

We describe now how we configured the AFS machine described in 2.4.3, "Installation of a Stand-Alone AFS Client on AIX" on page 52, to be a stand-alone AFS client.

2.4.4.1 Executing the AIX Kernel Extension Facility

First, you should run the following two commands:

```
/usr/vice/etc/dkload/cfgexport -a /usr/vice/etc/dkload/export.ext.nonfs  
/usr/vice/etc/dkload/cfgafs -a /usr/vice/etc/dkload/afs.ext
```

Such commands execute the AIX kernel extension facility to load AFS into the kernel dynamically and configure the machine to act only as an AFS machine and not as an NFS/AFS translator machine. More details on these two commands are explained in 1 on page 36.

2.4.4.2 Cache Manager Configuration

Next we created two new directories /afs and /usr/vice/cache by entering:

```
mkdir /afs  
mkdir /usr/vice/cache
```

When we configured the AFS client on the same machine where we had already configured the AFS server (see 2.4.2.6, "Cache Manager Configuration" on page 42) we found the two files /usr/afs/etc/ThisCell and /usr/afs/etc/CellServDB

already present, because we had previously executed the `afsConfigureServer.ksh` script file. It was the AFS server configuration that generated and put them in the right directory. In this case, when we have to configure a stand-alone AFS client on AIX, we have to manually create such files, or get them from the machine that acted as file server in our cell. We chose to get them using the File Transfer Protocol (FTP). The File Server Machine from which we got them was `rs600030.itso.ral.ibm.com` (see 2.4.2, “Configuration of the First AFS Machine on AIX” on page 29). We put them under the new paths `/usr/vice/etc/ThisCell` and `/usr/vice/etc/CellServDB` respectively. For more information about these two files and their contents, see 2.4.2.6, “Cache Manager Configuration” on page 42.

Next we created the `cacheinfo` file for the Cache Manager. We followed the same procedure described in 2.4.2.6, “Cache Manager Configuration” on page 42, and entered the command:

```
echo "/afs:/usr/vice/cache:100000" > /usr/vice/etc/cacheinfo
```

2.4.4.3 Starting the Cache Manager

We started the Cache Manager by launching the script file `rc.afsd.large` (see Figure 23 on page 44) that comes with the installation of IBM WebSphere Performance Pack in both the directories `/usr/vice/etc/dkload` and `/public/WebSphere/AFS`. As we explained in 2.4.2.7, “Starting the Cache Manager” on page 44, two other files would have been available to launch such a command: `rc.afsd.medium` (see Figure 24 on page 45) and `rc.afsd.small` (see Figure 25 on page 45). We chose `rc.afsd.large`, because it is more appropriate for large systems. We assumed that in our scenarios we would have a large number of users. Moreover, the AIX machine where we were configuring the stand-alone AFS client was powerful enough; it had 132MB of RAM and we had configured its cache partition in the `cacheinfo` file to be 100MB. For this reason, the file `rc.afsd.large` was more appropriate for our needs.

This was the command we issued to launch the Cache Manager on our stand-alone client machine:

```
/usr/vice/etc/dkload/rc.afsd.large
```

2.4.4.4 Clock Synchronization on AIX

Before you can authenticate yourself as the AFS user `admin`, it is necessary to perform another important operation. You should make sure that the internal clock on the AFS client is synchronized with the internal clock on the AFS server machine. If these two clocks give two different times, you cannot authenticate yourself.

There are several ways (including a manual setting) that you can use to adjust the internal clock on an AIX machine to be synchronized with the clock on another AIX machine, which in this case can be considered as a time server. We selected to use the `setclock` command. We entered the following on the AFS stand-alone client machine `rs600012`:

```
setclock -rs600030
```

Then we verified that its internal clock had been automatically synchronized with the clock of the File Server Machine `rs600030`.

Clock synchronization is very important. To ensure that the two clocks are always synchronized, we decided to have our AFS client machine synchronize its internal clock with the internal clock of the first AFS machine automatically every 10 minutes. To do this, we first entered the command:

```
crontab -e
```

This command allowed us to edit a file in vi edit mode. Such a file lists all the commands that need to be executed periodically, and allows you to specify the period as well. We added the following entry:

```
10 * * * * /usr/bin/setclock rs600030 > /dev/null
```

The first five fields in the above line indicate the period, specified through number of minutes, hours, days, months and years. For this reason, `10 * * * *` simply means that the `setclock` command must be executed every 10 minutes. Notice that this command also produces output, since it generates mail that is sent to the root user each time the command is executed. This explains the redirection to `/dev/null`.

Observation

Notice that clock synchronization is automatically executed on UNIX machines if the `afsd` daemon is launched without the `-nosettime` flag. Then in this case the `afsd` daemon picks up the time from one of the AFS server machines in the cell.

Use `-nosettime` on servers that also run the AFS client because they already use `runntp` to keep the time synchronized.

When a cell has multiple AFS server machines, they all should run the `runntp` command to synchronize their clocks with each other, except one of them, that runs the `runntp` command as well, but could use the `-localclock` flag. This machine picks up the time from an external time server, or it is a time server itself.

The `-localclock` flag tells the Network Time Protocol Daemon (NTPD) to use the local machine's internal clock as a possible source of the correct time in case a network partition separates the machine from the specified external time source(s). Cells connected to the Internet should not normally use this flag. In cells that experience frequent separations from the network (voluntary or otherwise), the `-localclock` flag should be used only on the System Control Machine.

The suggested installation of `runntp` is to have the System Control Machine run an instance of `runntp` like the following:

```
Instance runntp, (type is simple) currently running normally.
Process last started at Thu Nov 5 10:15:57 1998 (1 proc starts)
Command 1 is '/usr/afs/bin/runntp vice2.fs.andrew.cmu.edu clock-1.cs.cmu.edu
```

The above output is obtained by entering the command:

```
bos status server -long
```

One or more external time sources are specified on the `runntp` command line. All other AFS servers in the cell generally run `runntp` with no parameters.

Now we noticed that the default installation of the first AFS machine (see 2.4.1, “Installation of the First AFS Machine on AIX” on page 22) forces the AFS server to run `runntp` with the `-localclock` flag. We had only one server in our cell, but if you have multiple servers running `runntp` with the `-localclock` flag, the servers would not be synchronized, and also the clients would probably not, because they could synchronize their clocks to different servers. For this reason we have explained here how to synchronize the internal clock of an AFS client machine on AIX.

If, instead of following the procedure we have suggested, you want to adjust the installation, you can follow these steps:

1. Remove the current `runntp` instance that uses the `-localclock` flag:

```
bos shutdown server runntp
bos delete server runntp
```

2. On the System Control Machine, create the new instance using external time sources:

```
bos create server runntp simple "/usr/afs/bin/runntp vice2.fs.andrew.cmu clock-1.cs.cmu.edu
```

3. On other servers, create the new instance as follows:

```
bos create server runntp simple "usr/afs/bin/runntp"
```

2.4.4.5 Authenticating As admin

After this operation has been performed, you can, and you should, authenticate yourself as admin, by entering:

```
/usr/afs/bin/klog admin
```

When you are prompted for admin's password, type it and press the Enter key. This operation is necessary at this point to verify that the AFS client machine is able to connect the Authentication Database on the AFS server machine.

2.4.4.6 Writing the Boot File for the Stand-Alone AFS Client Machine

We also built the script file `/etc/rc.afs` to execute the AIX kernel extension facility and start the Cache Manager at each reboot. To edit such a file, you can use for example the `vi` text editor:

```
vi /etc/rc.afs
```

This is the file that we created:

```
#!/bin/ksh
echo 'loading AFS ... ' > /dev/console
cd /usr/vice/etc/dkload
./cfgexport -a export.ext.nonfs
./cfgafs -a afs.ext
/usr/vice/etc/dkload/rc.afsd.large
```

Figure 30. Contents of the File `/etc/rc.afs` on the Stand-Alone AFS Client AIX Machine

The considerations about this `rc.afs` file are the same assumptions we made in 2.4.2.14, “Writing the Boot File for the First AFS Machine” on page 49, about the `rc.afs` file on the first AFS machine (see Figure 26 on page 50). The only difference is that here you have a stand-alone client where the AFS server processes will not run. For this reason, you don't need to start the BOS Server, or

turn on the authentication checking, unlike you have to do on an AFS server and client machine.

Next, we edited the file `/etc/inittab` and we added the following entry:

```
rcafs:2:wait:/etc/rc.afs >/dev/console 2>&1 # Start AFS
```

Do not forget to make `/etc/rc.afs` executable. This can be done by entering the command:

```
chmod 744 /etc/rc.afs
```

Now, you can consider the configuration of your stand-alone AFS client completed.

2.4.4.7 After Rebooting the Stand-Alone AFS Client Machine

Reboot your machine and let the file `/etc/rc.afs` be automatically executed. Then, you should be able to determine whether the Cache Manager started correctly. The following command will help you see all the AFS client processes:

```
ps -ef | more
```

The output you received from that command will display all the processes running on your AFS stand-alone client machine. We selected only the processes that effectively depend on AFS and we show you them in the following figure:

```
root 3194 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 3452 1 0 Sep 28 - 0:24 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 4144 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11098 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11352 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11610 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 11868 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12126 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12384 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12642 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 12900 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13158 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13416 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13674 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 13932 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 14190 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 14448 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
root 14706 1 0 Sep 28 - 0:00 /usr/vice/etc/afsd -stat 2800 -dcache 2400 -daemons 5 -volumes 128
```

Figure 31. AFS Client Processes on the Stand-ALone AFS Client

Notice that the above figure is very similar to Figure 28 on page 52. As you can see, all the AFS client processes are generated by the `afsd` daemon.

You should also verify that your AFS client machine can access the AFS volumes `root.afs` and `root.cell`. To do this, we first accessed the directory `/afs` by entering the command:

```
cd /afs
```

Then we verified that the command `ls -l` listed the two directories `.itso.ral.ibm.com` and `itso.ral.ibm.com`.

2.4.5 Installation of a Stand-Alone AFS Client on Windows NT

We have already mentioned that on Windows NT an AFS machine can be only a stand-alone AFS client, since the AFS server component cannot be installed on this platform.

We installed the File Sharing client component of IBM WebSphere Performance Pack for Windows NT on a PC IBM Model 750, with 128MB of RAM, CPU of 166 MHz, and 1.5GB of hard disk. We had previously installed this machine with Windows NT Version 4.0, and we had also applied Service Pack 3. The host name for this machine was WTR05195 and the IP address for its token-ring network adapter was 9.24.104.223. This machine was part of a workgroup named WTRDM. We performed another successful installation also on another Windows NT machine, having very similar features, except for the fact that it had only 64MB of RAM.

It is important to mention also that we had set the paging space on our Windows NT system to be 328MB. 128MB of virtual memory was available on the C drive and 200 on the D drive, as shown in the following screen:

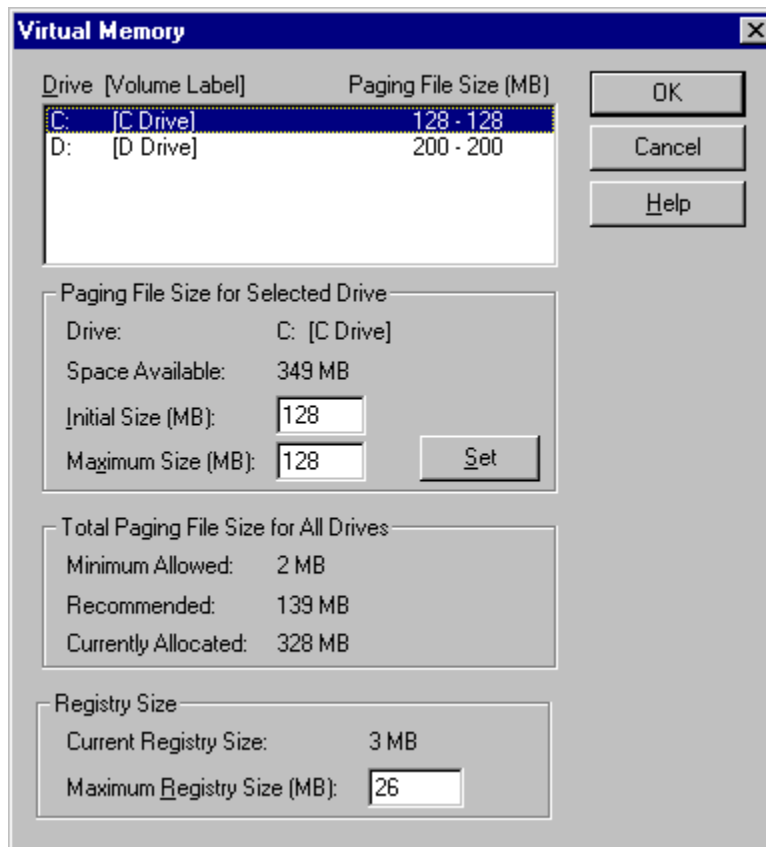


Figure 32. Virtual Memory Configuration on Our Windows NT AFS Client Machine

The Windows NT installation program for each component of IBM WebSphere Performance Pack makes use of the Java InstallShield's setup class. You are required to pre-install the Java Virtual Machine (JVM) Version 1.1 or higher, which is incorporated into the Java Development Kit (JDK) Version 1.1 or higher. Actually you don't need the full JDK, you only need its subset known as Java

Runtime Environment (JRE), which contains just the JVM, Java platform core classes, and supporting files. In other words, the JRE is the smallest set of executables and files that constitute the standard Java platform and it contains only the run-time part of the JDK: no compiler, no debugger, no tools. The CD of IBM WebSphere Performance Pack ships with the JDK 1.1.5 for Windows NT, found in the directory JDK\NT. This version of the JDK worked very well for us, but if you want to install a later version, you can download it from the JavaSoft Web site <http://www.javasoft.com>.

The JDK 1.1.5 installation for Windows NT was very easy, so we skip its description here. For a detailed description, you can see the IBM redbook *Internet Security in the Network Computing Framework*, SG24-5220.

To install the File Sharing client component of IBM WebSphere Performance Pack, run the setup.exe program from the CD-ROM installation directory, named NT. To do this, from the Start menu, select **Run...**, then click on **Browse...** and open the setup.exe program located in the NT directory, as shown in the following figure. (Notice that in our example, drive E was assigned to the CD-ROM drive.)

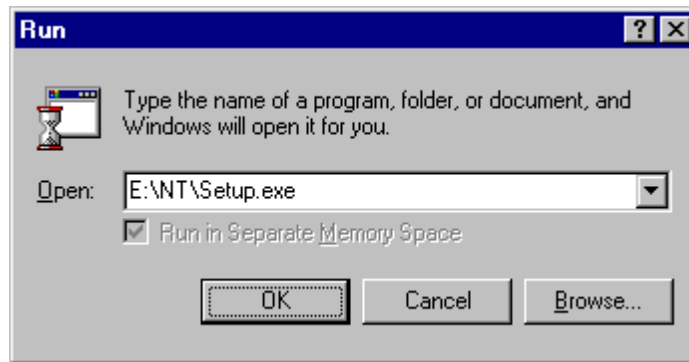


Figure 33. Installation Program's Name and Path

Click **OK** to run the installation routine. It will start to find all the JVMs installed on your system. Since we had already installed the JDK 1.1.5, the following window appeared:

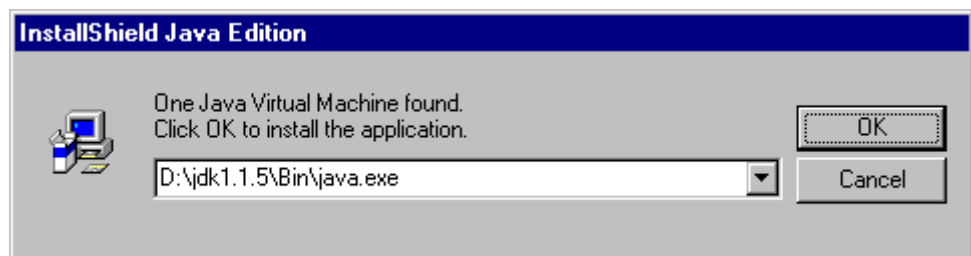


Figure 34. Selection of Java Virtual Machine Version

We clicked the **OK** button and the Welcome window was displayed. After selecting the **Next** button, you are required to agree to all items of the software license agreement. If you agree, check the box **Accept all terms of the license** then click **Next**. You will get another window displaying the IBM WebSphere Performance Pack Version 1.0 readme file. We suggest you to take a look at that file as it contains interesting information about the product.

Click on **Next** and a dialog will be displayed where you can select the language only for the Load Balancing component, even if you have not yet specified that you want to install just that component:



Figure 35. Load Balancing Component Language Selection

After you click **Next**, you are prompted to enter the IBM WebSphere Performance Pack installation directory, which by default is named WebSphere, and we had the setup program create it on the D drive, as shown in the next window:

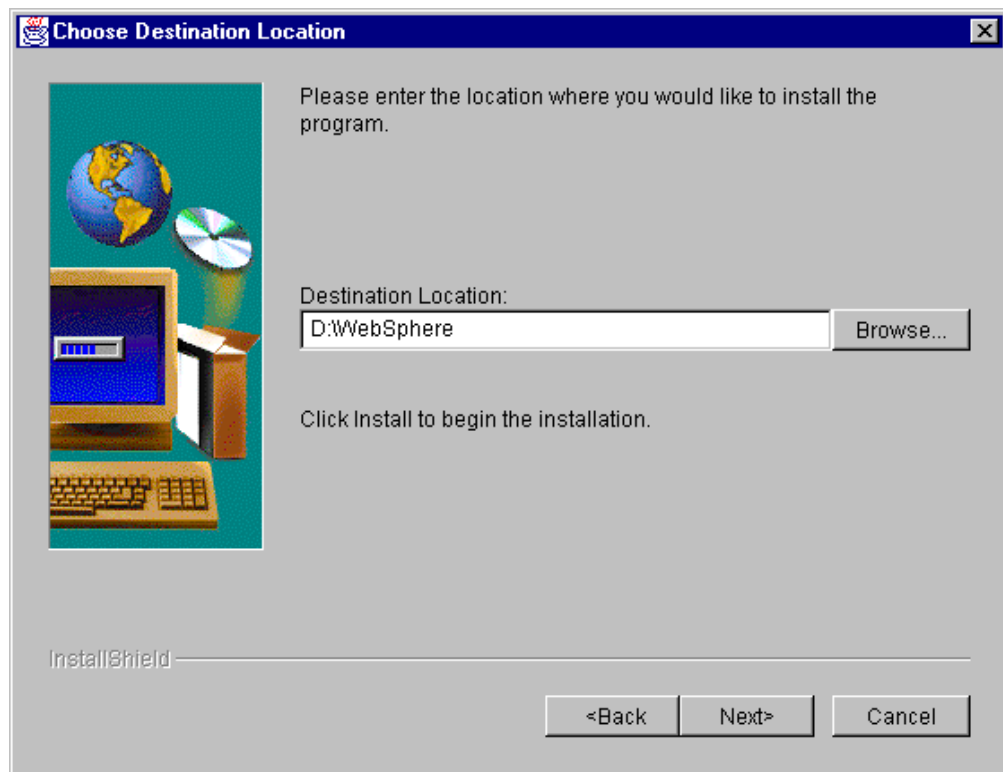


Figure 36. Choose Destination Location

Even if the above window invites you to click **Install** to continue with the installation, such a button does not exist. You should click on **Next** instead.

Select the IBM WebSphere Performance Pack component that you wish to install on your Windows NT platform. Note that you cannot select the check box **File Sharing Server**, since such a component is not available on Windows NT installation. We selected only the **File Sharing Client** component, as shown in the following figure:

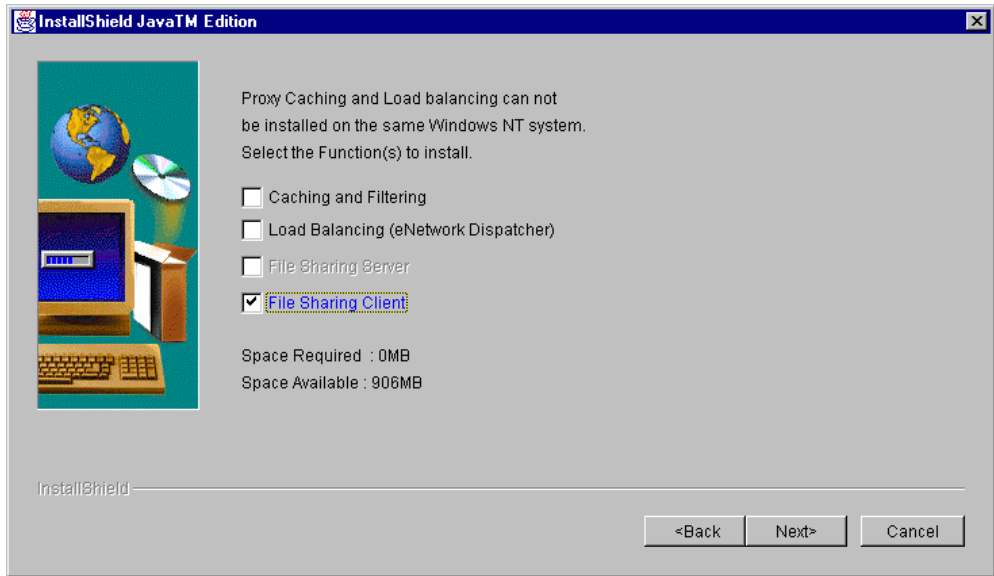


Figure 37. Selection of File Sharing Client component

The File Sharing client component on Windows NT is compatible with the Caching and Filtering component and with the Load Balancing (eNetwork Dispatcher) component.

We clicked on **Next** and the following window appeared, giving us the possibility to replace the current version of any product that had already been installed:



Figure 38. Replacing the Current Version of Any Product Already Installed

The default selection in the above figure is **Yes**, but we preferred to select **No**, because we had not installed any other product on our Windows NT platform, except the operating system itself and the Service Pack 3. Then we clicked on the **Install** button. At this point, the installation should proceed and complete in a short while, and you should see the following window, which offers you the possibility to pass directly to the AFS client configuration:

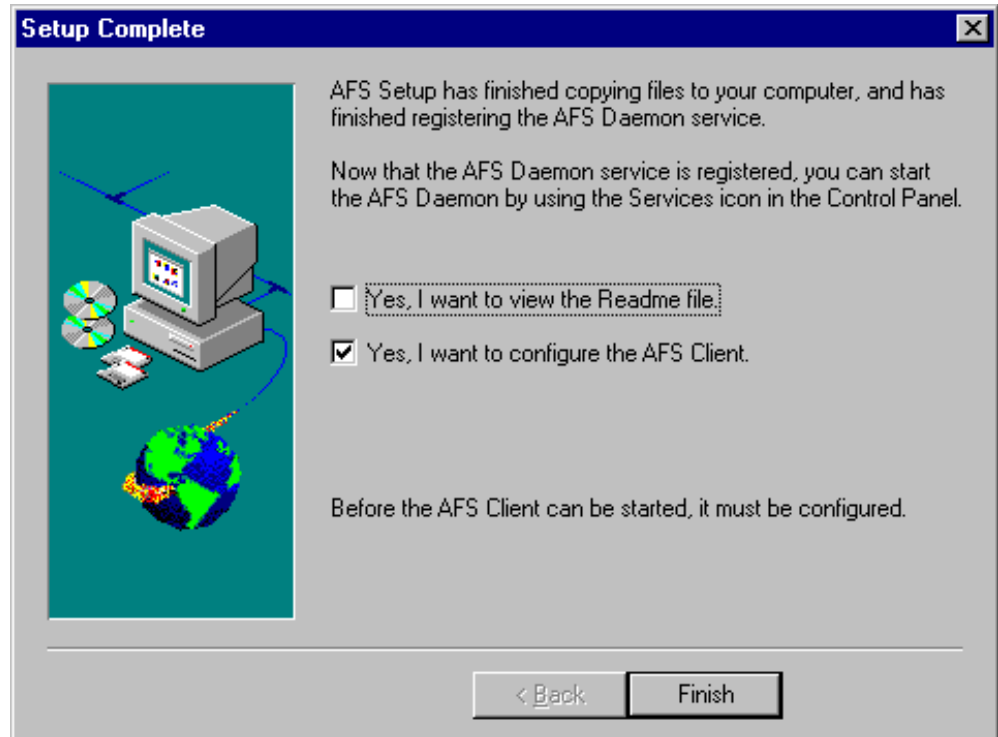


Figure 39. AFS Client Setup Complete

2.4.5.1 How to Recover an Installation Problem

Things could be not as simple as shown. We performed several installations of the File Sharing client component and we saw that often the installation does not complete successfully. We show you how to proceed if instead of Figure 39 on page 63, you see the following error screen:

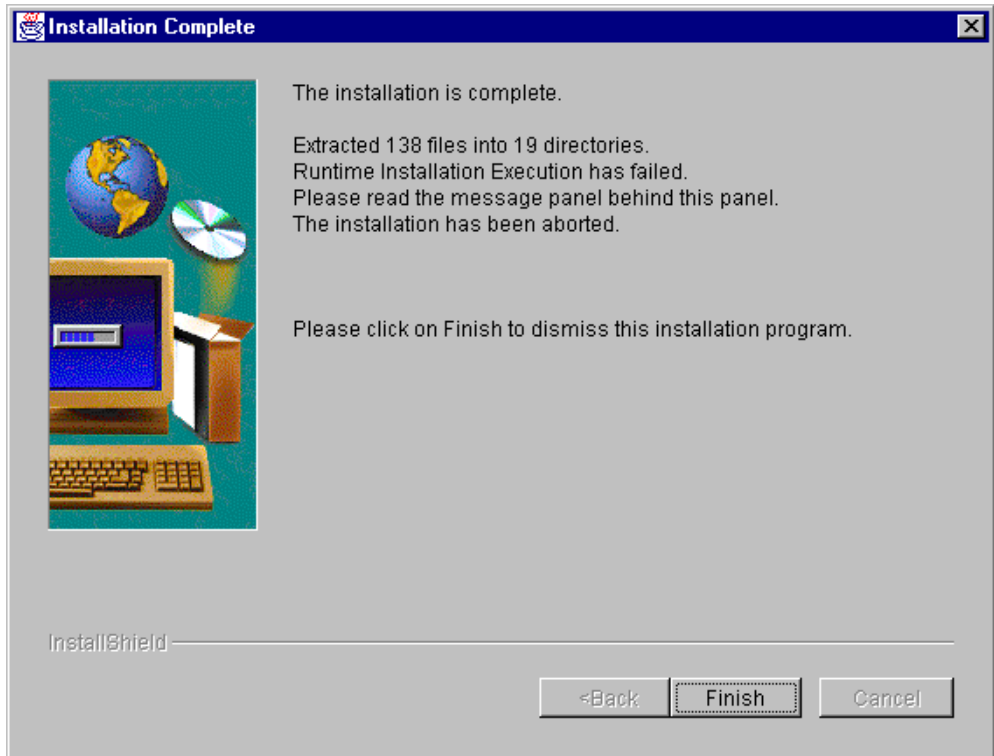


Figure 40. The Runtime Installation Execution Has Failed

Behind this window, another panel will be brought up:

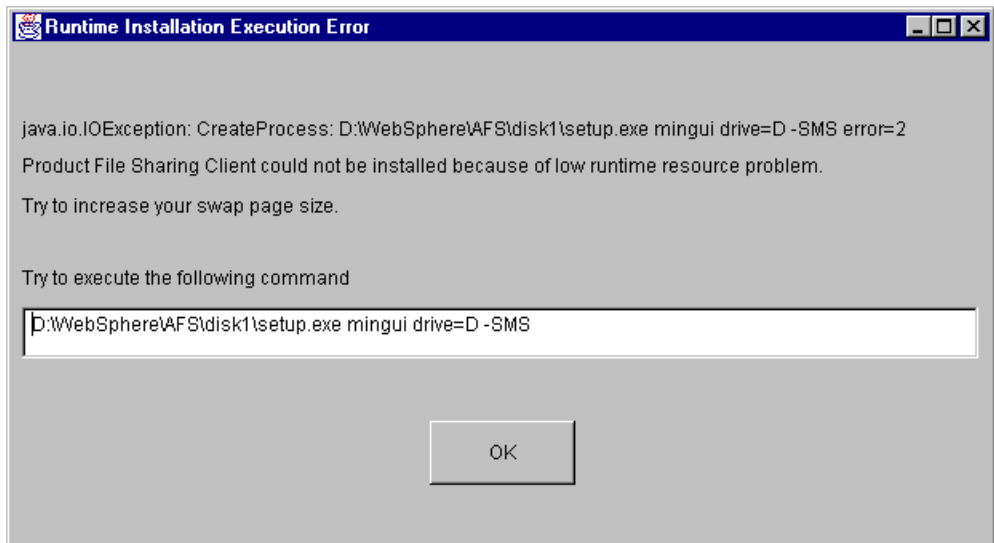


Figure 41. Runtime Installation Execution Error Window

We followed the directions on the above screen. We opened a Command Prompt window and entered the following command:

```
D:\WebSphere\AFS\disk1\setup.exe mingui drive=D -SMS
```

After the execution of the above command, the installation completed successfully, but it still required some basic configuration steps, which are incorporated as part of the installation process.

2.4.5.2 Basic Configuration

We had the possibility to proceed with the AFS client configuration, as shown in Figure 39 on page 63. We checked the option to configure the AFS client and chose the **Finish** button. The AFS Client Configuration panel was brought up and we filled all its fields according to our AFS configuration:

- We set Cell Name to `itso.ral.ibm.com` (see 2.4.2, “Configuration of the First AFS Machine on AIX” on page 29).
- We set the size of the local cache to be 100MB. We had set the same value also for the cache of the AFS client on AIX (see 2.4.4.2, “Cache Manager Configuration” on page 53). As we have already explained, the cache on an AFS client machine is used to store local copies of all the files that are brought up from File Server Machines. The cache size must be at least 1MB and cannot exceed the total virtual memory paging space available on the machine. In fact, as soon as the Cache Manager is launched, the AFS client reserves virtual memory paging space equal to the cache size. By default this value is set to 20480KB. Since the virtual memory paging space on our system was 328MB (see Figure 32 on page 58), we could safely set the cache size to 100000KB.
- We accepted the default value of 4096 bytes for the Page Size field. This value is used on the client machine by the Cache Manager, which gets data from the AFS server and caches it in the local disk. The Cache Manager stores data in sections of the cache called pages. The page size must be at least 2KB and a power of 2. The default value is 4KB and we kept it. If you want to change this value, you should consider the following:
 - A small page size can cause performance problems, because the Cache Manager spends more time managing several small units of data than it does managing fewer larger units.
 - A large page size decreases the number of sections in the cache for data storage. This can result in wasted cache, because each page can store data from only one file. If the size of a file is less than the page size, the remainder of that page goes unused.
- The field Stat Entries displays and specifies the number of files in the cache about which the Cache Manager records status information. We kept for this field the default value of 1000 entries.

The following figure shows the AFS Client Configuration window after we completed our selections:

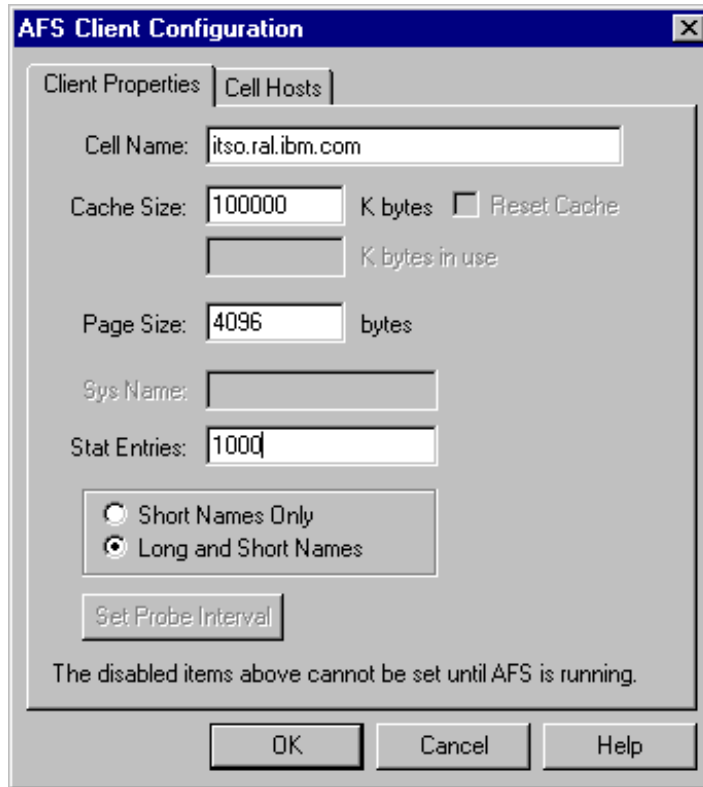


Figure 42. AFS Client Configuration Panel - Client Properties

After finishing our selections, we clicked on the **OK** button, and the following error message was brought up:

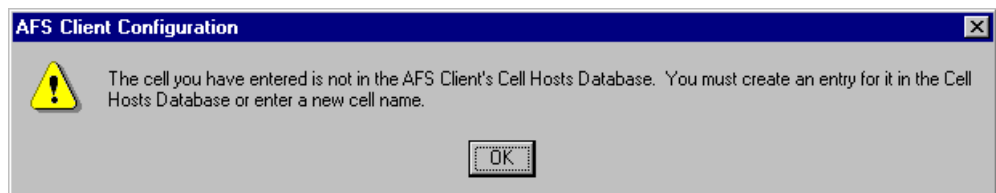


Figure 43. AFS Client Configuration Message

This is normal because the AFS client comes with a predefined cell database, where the cell name `itso.ral.ibm.com` we had entered was not included. We clicked **OK** in the above warning message window, then we went back to Figure 42 on page 66 and we clicked on the **Cell Hosts** tab. A screen similar to the following was displayed:

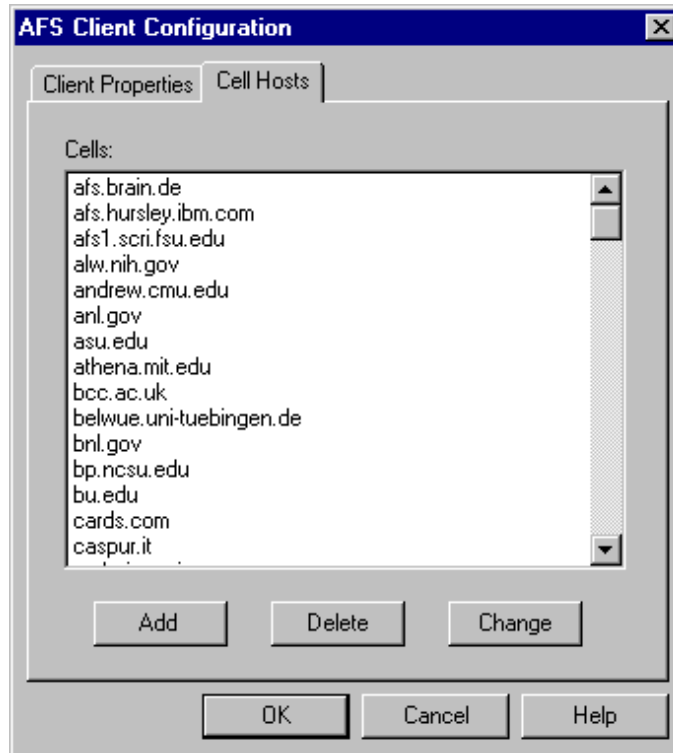


Figure 44. AFS Configuration Client Panel - Cell Host Choices

This list includes the cell names of all the companies that accepted their cell to be notified during the configuration of the File Sharing client component of IBM WebSphere Performance Pack. Since `itso.ral.ibm.com` was not one of those cell names, we had to add it, and to do this we clicked on the **Add** button. The Add Cell window was displayed, and we entered `itso.ral.ibm.com` in the Cell Name field, `rs600030.itso.ral.ibm.com` in the Name field of the Servers section, and `9.24.104.97` in the IP Address field, as shown in the following figure:

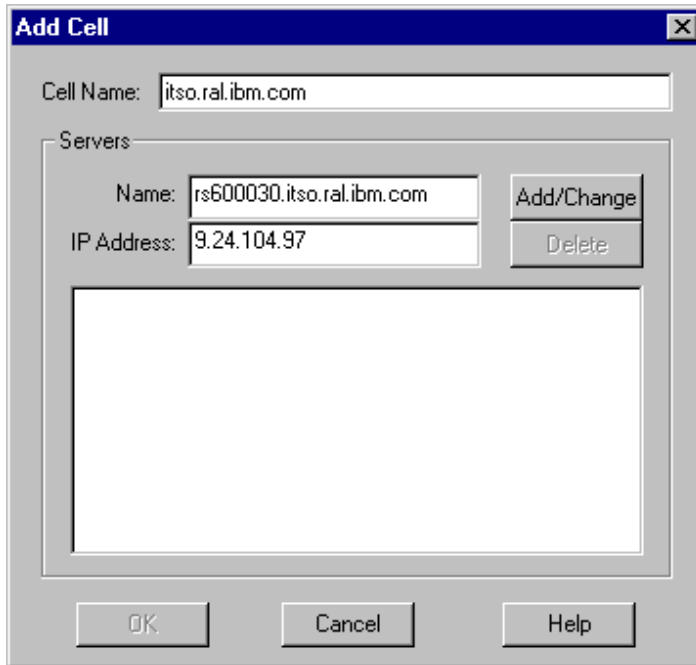


Figure 45. Add Cell Panel while Entering Data

Notice that in the Name field it is recommended to enter the fully qualified name of the machine, rather than the simple host name.

Then we clicked on **Add/Change** to have the configuration process accept these values, as shown next. Notice that rs600030 was the host name of our first AFS machine, while 9.24.104.97 was its IP address, as you can verify in 2.4.1, "Installation of the First AFS Machine on AIX" on page 22.

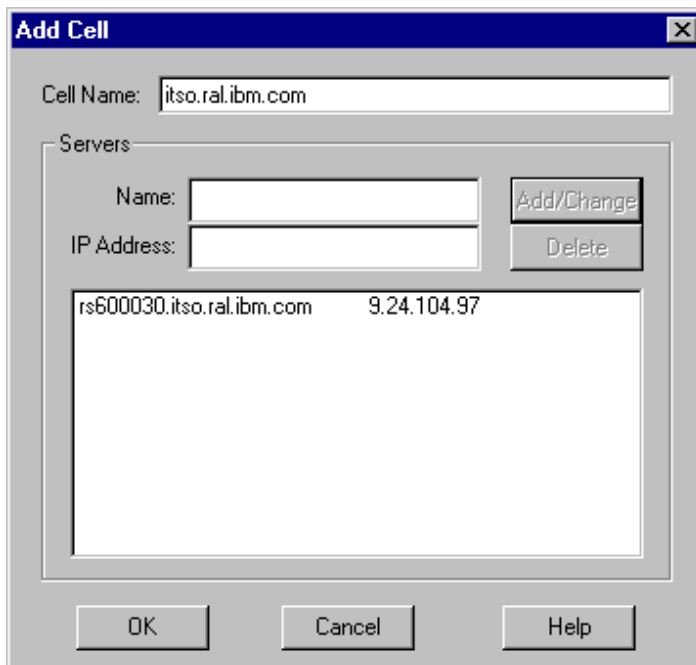


Figure 46. Add Cell Panel after Data Have Been Accepted

Here we clicked **OK**. At this point we were informed that the AFS client installation was complete:

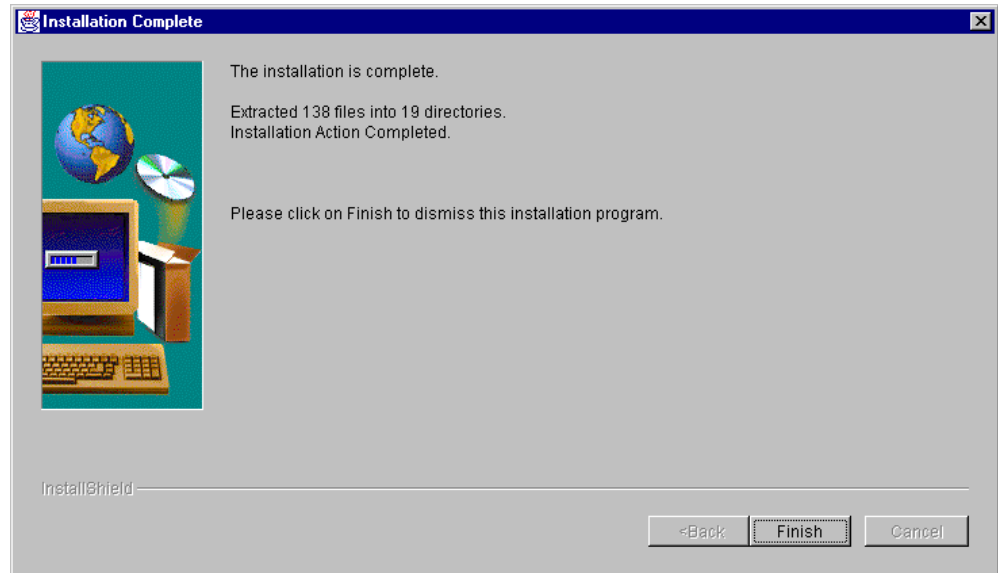


Figure 47. Installation Complete

We clicked on **Finish** to complete the installation.

2.4.6 AFS Patch Installation on Windows NT

The installation of the AFS client on Windows NT 4.0 requires a patch to be installed immediately after the AFS client itself. It is not necessary to restart the machine at this point. You can download the current patch for Windows NT 4.0 from the Transarc Web site <http://www.transarc.com>.

AFS Patch for Windows NT

In the book *IBM WebSphere Performance Pack for Multiplatforms - Getting Started* (shipped with the product but not separately orderable), we read that you should download and install the AFS Client Patch 8 for Windows NT 4.0, since it was the current patch when that manual was written. When we accessed the Transarc Web site, we found that the AFS Client Patch 9 was available. We downloaded and installed it, without finding any problems. For this reason, we recommend that you always download the current available patch, so that all the known problems are fixed.

The AFS Client Patch 9 for Windows NT 4.0 that we downloaded from the Transarc Web site came with a file named `afs34P9.exe`. Noticed that this file must be in a local disk to be launched. It cannot be launched from a remote drive. When we launched it, we saw the following screen:

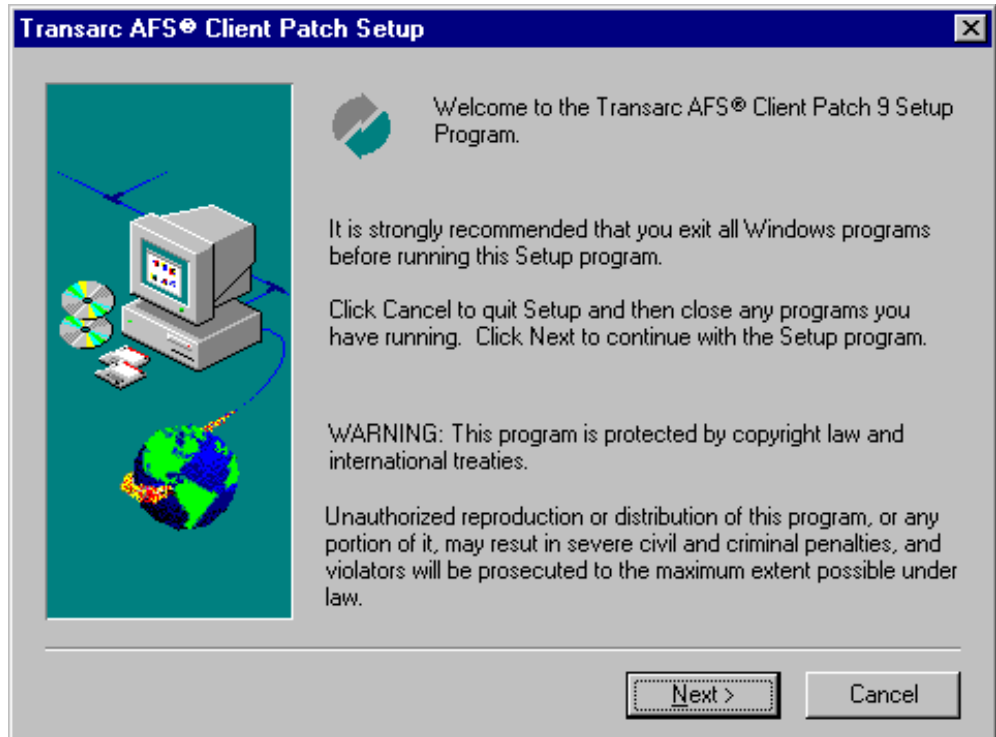


Figure 48. Transarc AFS Patch

We clicked on **Next** and the following window was brought up:

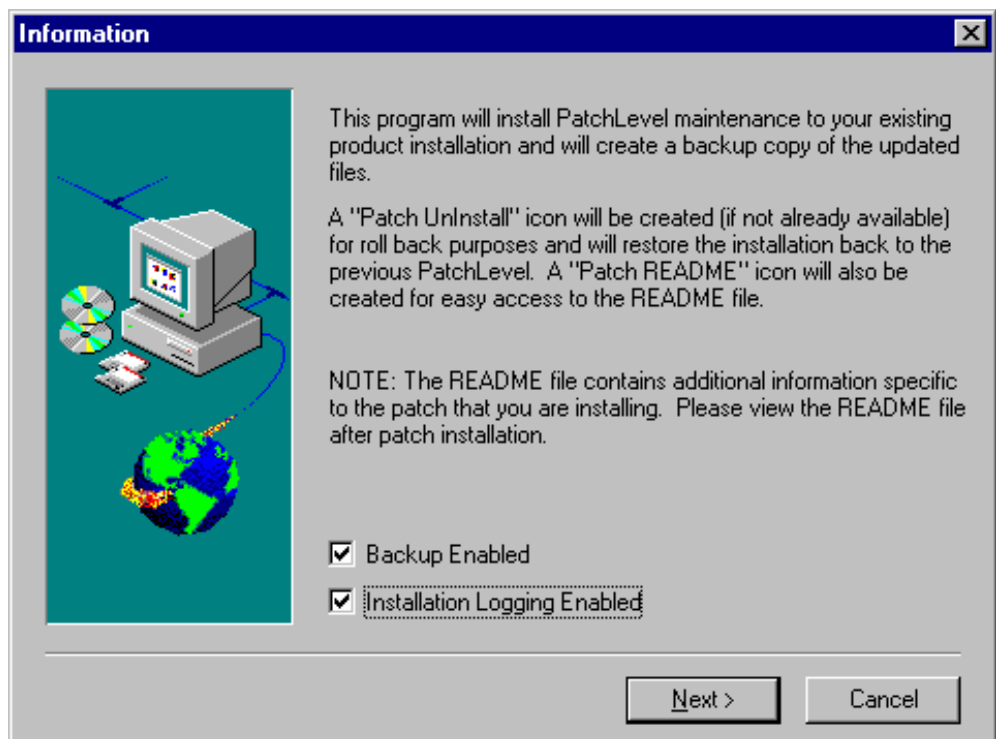


Figure 49. Information Panel

We accepted the default values, as shown, and then clicked again on **Next**. A warning message appeared at this point, which we show in the next figure:

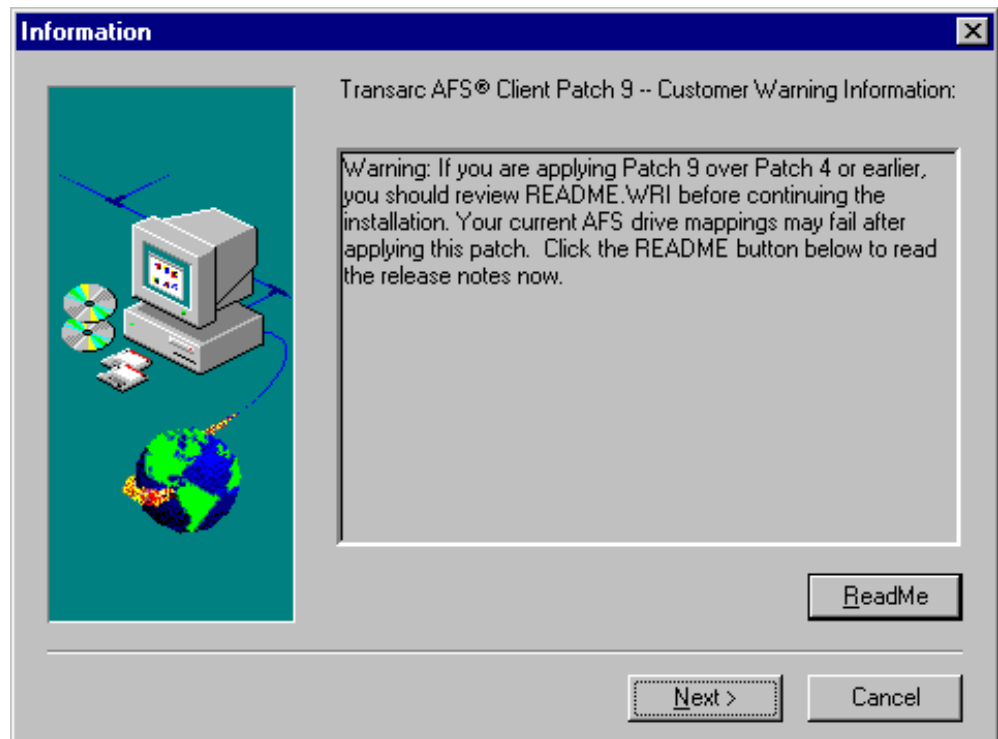


Figure 50. A First Warning Message During the Installation of the Patch 9

After this message, if you click on **Next**, the installation starts. During the installation a second warning message is brought up:

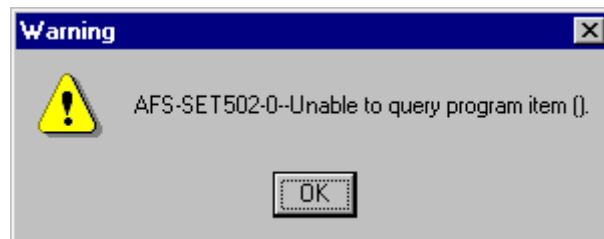


Figure 51. Second Warning Message during the Installation of the Patch 9

We ignored both the warning messages and then we completed the installation of the AFS Client Patch 9, as shown in the following screen:

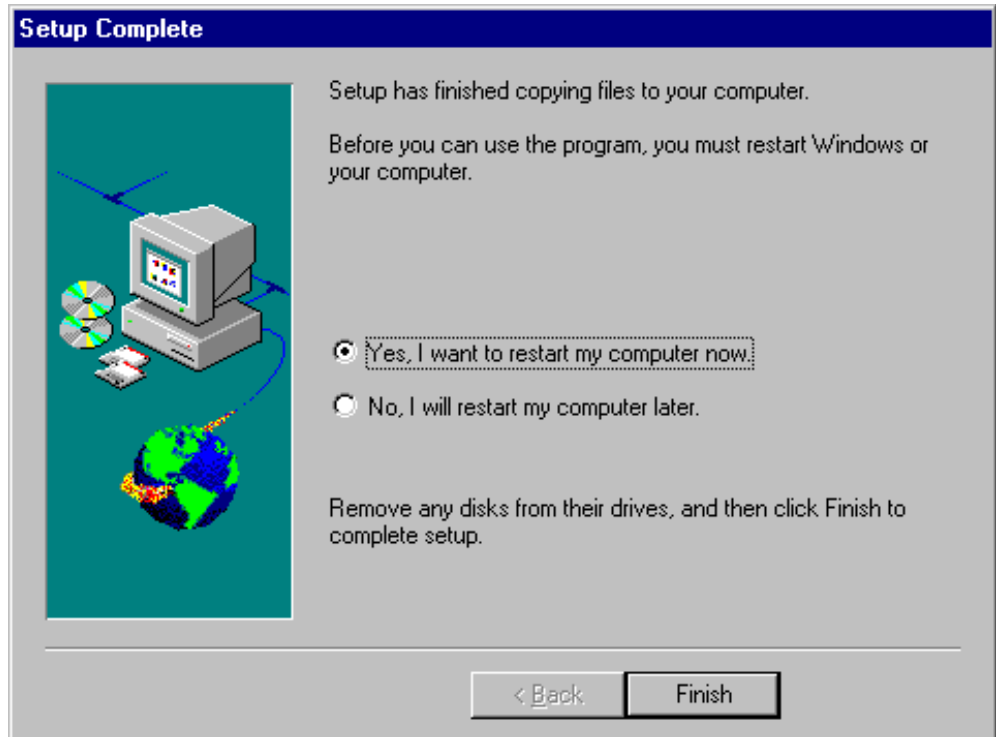


Figure 52. Setup Complete Panel

Notice that at this time it was necessary to restart the computer before proceeding.

2.4.7 Configuration of a Stand-Alone AFS Client on Windows NT

In this section, we show the steps to configure a stand-alone AFS client on Windows NT. The machine here is the same as in 2.4.5, “Installation of a Stand-Alone AFS Client on Windows NT” on page 58 and 2.4.6, “AFS Patch Installation on Windows NT” on page 69.

2.4.7.1 AFS Daemon Startup

After clicking on **Finish** in Figure 52 on page 72, the machine rebooted. After that, we noticed that the AFS daemon had automatically started. The Cache Manager is a process that allocates a lot of memory. As we said, the AFS client reserves virtual memory paging space equal to the caching size. For example, in our case, since we had set a caching size equal to 100MB, we noticed that as soon as the Cache Manager started, it allocated about 100MB of RAM of virtual memory, as the Windows NT Task Manager shows in the following screen:

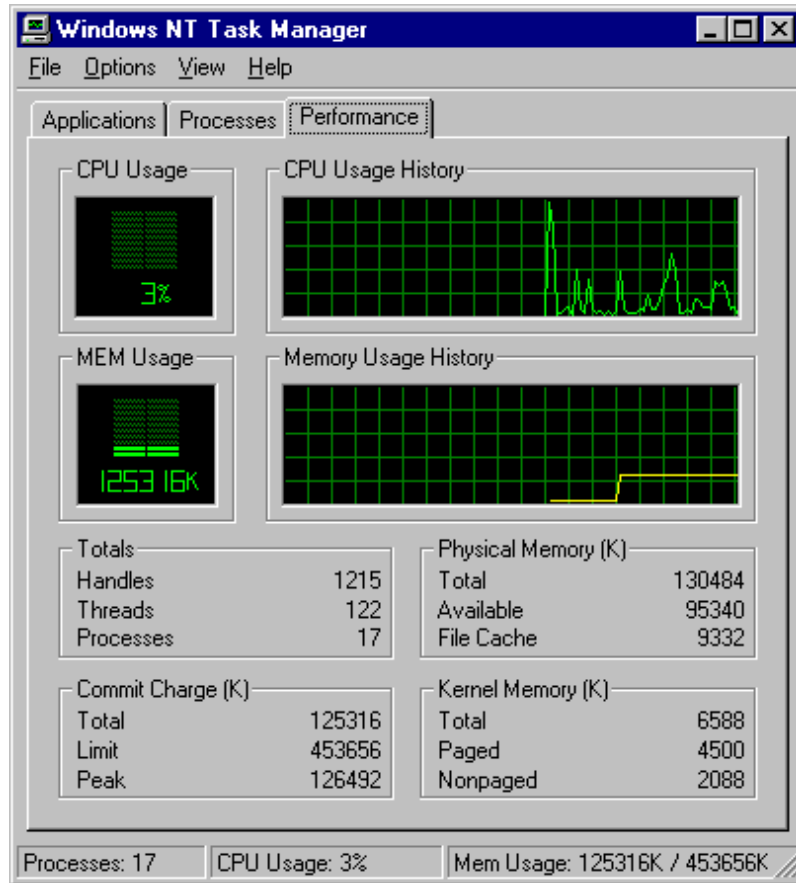


Figure 53. Windows NT Task Manager As Soon As the Cache Manger Starts

We decided to modify the startup mode and to set it to manual, in order to allocate such a big amount of memory only when effectively needed. We opened the Services window from the Control Panel of Windows NT, selected **Transarc AFS Daemon**, clicked on **Startup...**, selected **Manual** and then chose the **OK** button.

When the Startup mode for the Transarc AFS daemon is set to Manual, the Services window should appear as shown in the following window:

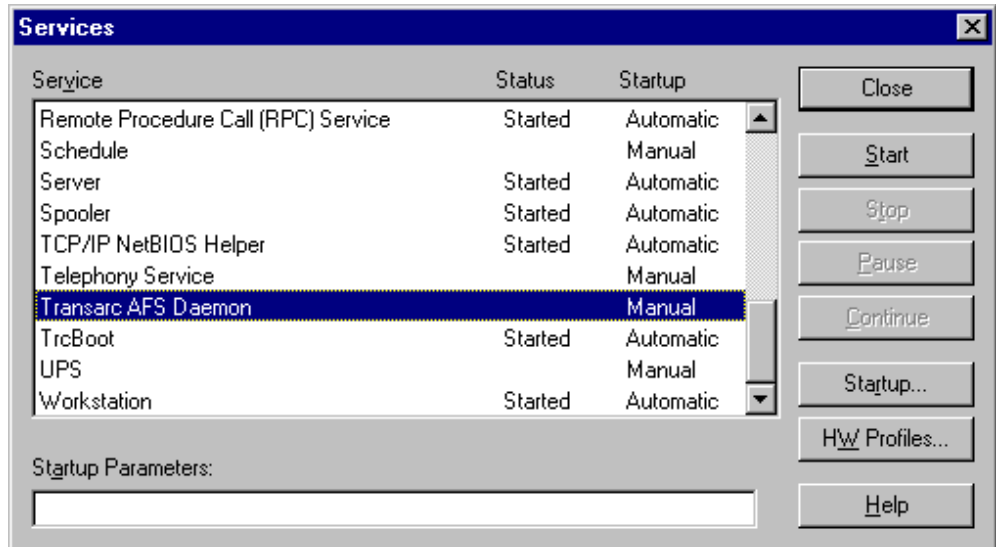


Figure 54. Transarc AFS Daemon Startup Type

Now we describe how to authenticate from a Windows NT AFS client machine. In this example, we show you how the AFS user admin can authenticate from the Windows NT AFS client machine that we have just installed and configured.

First, it is necessary that the AFS daemon is active. If that process starts automatically at the machine's reboot, you do not have to do anything. If you chose a manual startup, as we did, you should select **Transarc AFS Daemon** from the Service panel and click on **Start** (see Figure 54 on page 74). The status of this process should change to read *Started*. Another confirmation that the AFS daemon is now running, comes from the Services section of the Windows NT Diagnostics window, which can be accessed from the Start menu by selecting **Programs, Administrative Tools (Common)** and finally **Windows NT Diagnostics**, as shown in the following picture:

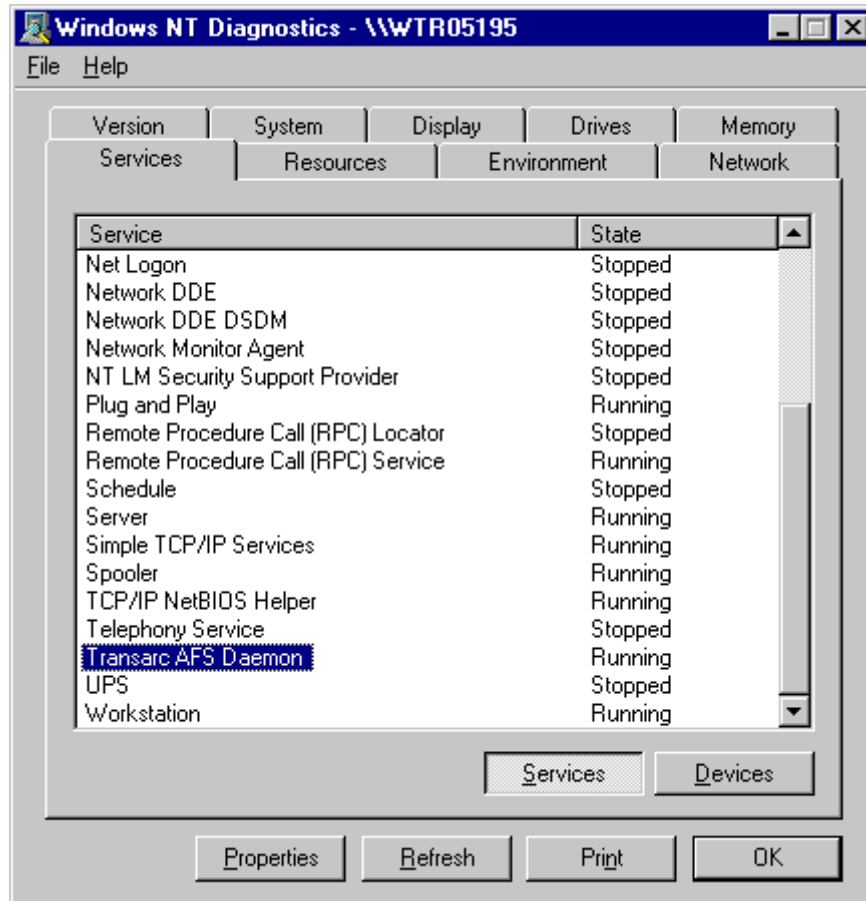


Figure 55. Control If AFS Client Process Is Running

2.4.7.2 Mounting the AFS File System

Next, in order to complete the steps required for AFS authentication, we mounted the AFS file system on rs600030, our first AFS machine, to a drive letter on our Windows NT machine. To do this, from a Command Prompt window, you should enter a command having the following syntax:

```
net use drive: \\hostname-afs\username
```

We replaced the variables with the actual values appropriate to our Windows NT system:

- We chose `H` as the value of the `drive` variable.
- We replaced `hostname` with `wtr05195`, which was the host name of our Windows NT machine.
- We typed `all` in place of `username`.

So, the above command in our platform became:

```
net use H: \\wtr05195-afs\all
```

You could also map a network drive to AFS in two other ways:

- From a Windows NT Explorer window, select **Tools** and then **Map Network Drive....**

- From the File Manager window (which can be opened launching the `winfile` command at a command prompt), select **Disk** and then **Map Network Drive....**

In both cases, the Map Network Drive window is displayed. We filled all its fields as indicated in the following screen:

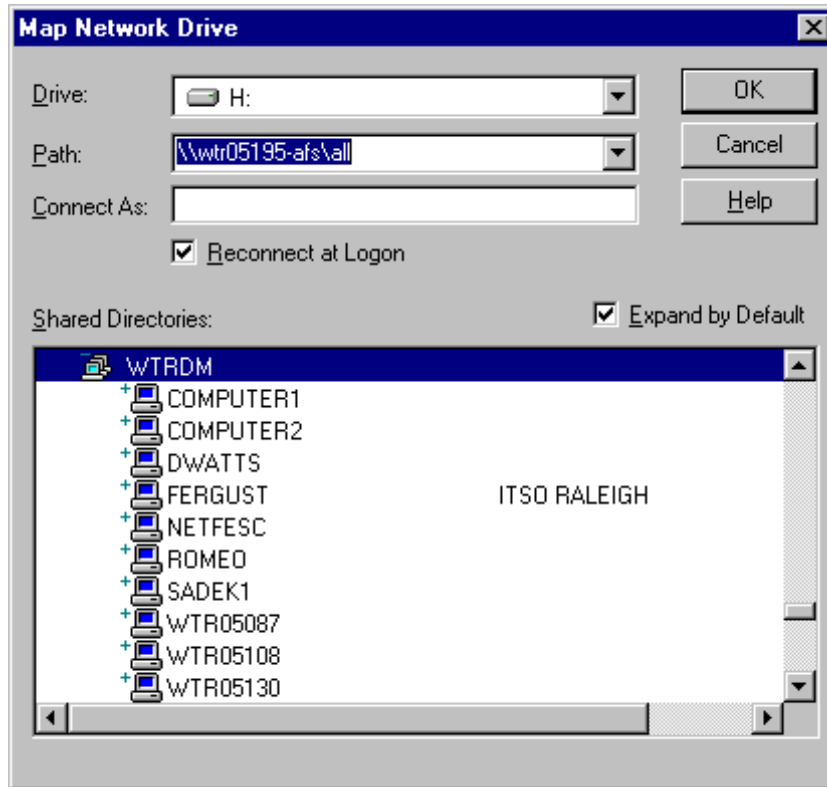


Figure 56. How To Mount the AFS File System

Notice that the check box **Reconnect at Logon** is marked. Since this mapping requires the AFS daemon to be active, if the AFS daemon does not start automatically at the machine's reboot, you will receive an error message while your system attempts to re-map the AFS file system.

After this step, you can already see and access (as an unauthenticated user) the AFS file system mounted on your AFS client machine:

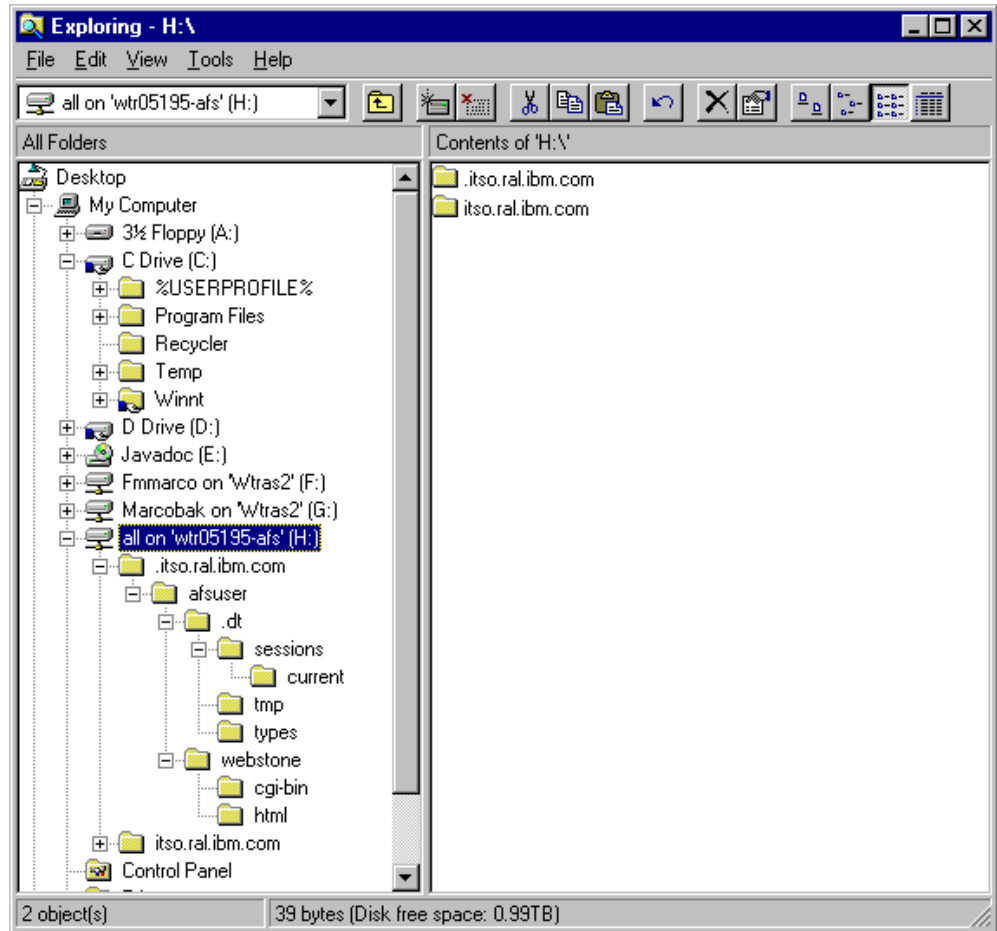


Figure 57. Accessing the AFS File System

Since you are not authenticated yet, you only receive permissions that have been granted to the members of the group system:anyuser, as we see in 2.5, “Creation of an AFS User” on page 85.

2.4.7.3 Clock Synchronization on Windows NT

Another important task is to control that the clocks on the File Server Machine and the AFS Windows NT client machine are synchronized. This is a pre-requisite to the AFS authentication. If the two clocks are not synchronized, the AFS authentication does not take place and an error message is displayed. It is absolutely necessary that the two clocks be synchronized, or you will not be able to authenticate to AFS.

The synchronization of the two clocks can be obtained in several ways. We even tried to do it manually, and our experiment was successful, but it took a long time and we discourage you to proceed with a manual synchronization, especially for systems with many Windows NT machines. Instead, you should use a program that automatically synchronizes the clock of your Windows NT AFS machine with the clock of the AFS server. There are several products that perform this function. We downloaded from <http://www.winfiles.com> a freeware program named NetDate Version 3.1415.

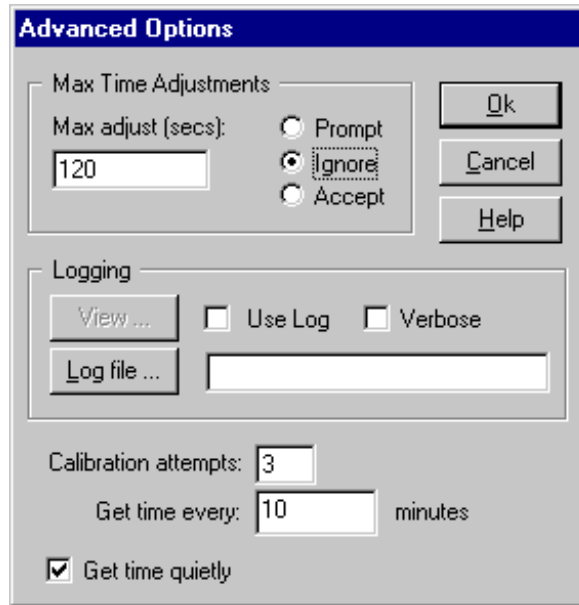


Figure 59. Advanced Configuration Panel

We made a test, moving the time one hour back on the Windows NT machine, and we saw that after a while the clocks on the two machines were synchronized again, since the Windows NT internal clock had modified its time according with the time given by the time server machine.

2.4.7.4 AFS Port Configuration on the AFS Server Machine

At this point, you are not ready yet to start the authentication process. We tried to do it, but every time we got the following error message, displayed on a pop-up window:

```
Authentication Server was unavailable.
```

Why this error message? If your Authentication Server process, `kaserver` (see 4 on page 37), is running on a SunOS, Solaris, or AIX 4.1 (or later) machine and you are seeing the above error message when attempting to authenticate with an AFS Client for NT, you need to make some modifications to the `/etc/services` file on your File Server Machine.

The following specifically demonstrates the default Kerberos values that we found on AIX 4.3. The default values for SunOS and Solaris can be different, but should still be changed to the correct values for use with AFS. When the `kaserver` starts, it checks the `/etc/services` file for entries for `kerberos` and `kerberos5`. If there are such entries in this file, then the `kaserver` listens on these ports for Kerberos authentication requests. If there are no entries in the `/etc/services` file for `kerberos` and `kerberos5`, then the `kaserver` process uses its known default ports for these services, which are:

```
kerberos 750
kerberos 88
```

It appears that AIX 4.3 machines have different values for `kerberos` than the ones listed above. The default `/etc/services` file for AIX 4.3 has the following entries:

```
kerberos 88/tcp # Kerberos
kerberos 88/udp # Kerberos
...
...
kerberos-adm 749/tcp # kerberos administration
kerberos-adm 749/udp # kerberos administration
rfile 750/tcp
loadav 750/udp
```

We made the following modifications:

```
kerberos5 88/tcp # Kerberos
kerberos5 88/udp # Kerberos
...
...
kerberos-adm 750/tcp # kerberos administration
kerberos-adm 750/udp # kerberos administration
rfile 750/tcp
loadav 750/udp
```

After you complete these modifications, restart the kserver process. We entered the following command which, as we have already explained, restarts all the server processes, including the BOS Server itself:

```
bos restart rs600030 -bosserver -cell itso.ral.ibm.com
```

To confirm that our changes had taken effect, we issued the command:

```
cat /usr/afs/logs/AuthLog
```

The output we got confirmed that our changes had taken effect:

```
kerberos/udp is unknown; check /etc/services. Using port=750 as default
kerberos5/udp port=88
Starting to listen for UDP packets
start 5 min check lwp
Starting to process AuthServer requests
```

Also the `netstat -an` command gave us interesting output. We entered:

```
netstat -an | grep 750
```

We got output similar to the following, which demonstrated that the Authentication Server kserver was listening on the correct port 750:

```
udp4      0      0  *.750          *.*
```

Port 7004

You probably noticed that we had not made any modifications to the file `/etc/services` when we had to authenticate as the AFS user `admin` from the stand-alone AFS client on AIX (see 2.4.3, “Installation of a Stand-Alone AFS Client on AIX” on page 52). The reason for this is that there are other two entries in the `/etc/services`:

```
afs3-kaserver 7004/tcp # AFS/Kerberos Auth. Service
afs3-kaserver 7004/udp # AFS/Kerberos Auth. Service
```

The AFS client on AIX uses the port 7004 for the authentication process, unlike an AFS client on Windows NT, which uses the port 750.

For this reason, when using AFS clients on AIX only, it is not necessary to modify anything in the `/etc/services` file.

For further details on the ports used by AFS, we recommend the document *Connecting Enterprise Data to the Web using WebSphere*, which you can find at <http://www.software.ibm.com/ebusiness/buzz.html>. The information provided on that document is very useful to understand what ports and protocols you need to enable in order to operate AFS through an Internet firewall.

2.4.7.5 Authenticating As admin

At this point, we were able to proceed with the authentication. On our Windows NT AFS client, we executed the command `afs_auth`. The file `afs_auth.exe` was located in the directory `D:\Transarc\AFS\AFSCClient\PROGRAM`. You can either change directories and launch that command from a command line or create a shortcut on your Windows NT desktop for the `afs_auth.exe` file and then double-click on its icon.

As a result, we saw the following screen:

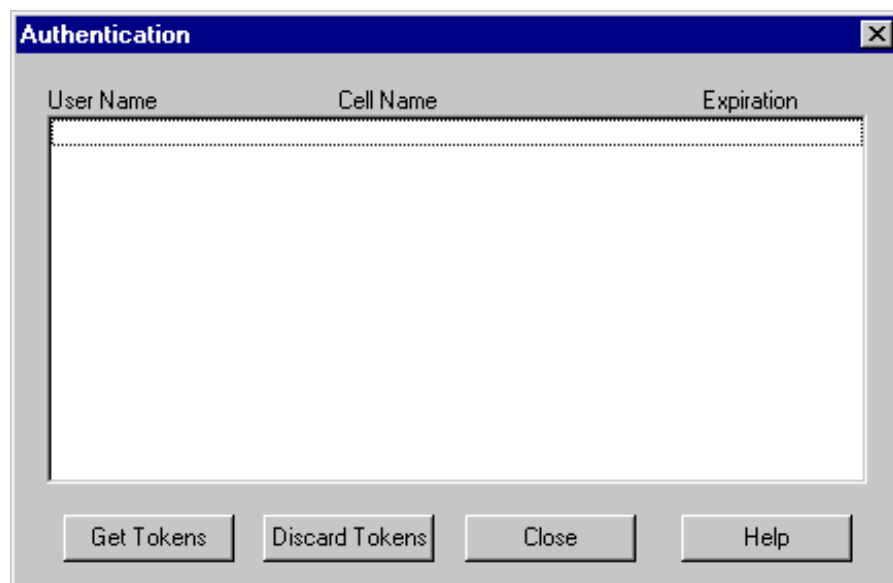


Figure 60. AFS Authentication Panel

We clicked on the **Get Tokens** button. On the dialog box that appeared, we filled the fields with information related to the admin AFS user that we had created at install time (see 2.4.2, “Configuration of the First AFS Machine on AIX” on page 29), as shown in the following screen:

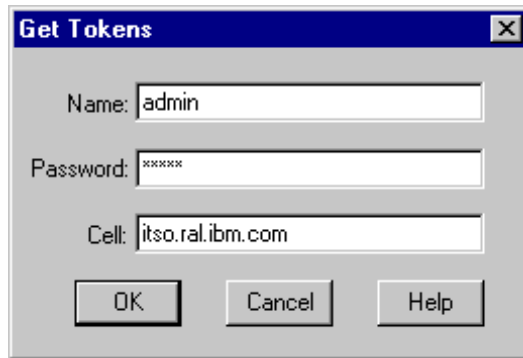


Figure 61. Tokens Panel

Notice that we had to fill in only the fields Name and Password, since the cell name was automatically displayed in the Cell field. As soon as we chose the **OK** button our authentication was confirmed, as shown in the following window:

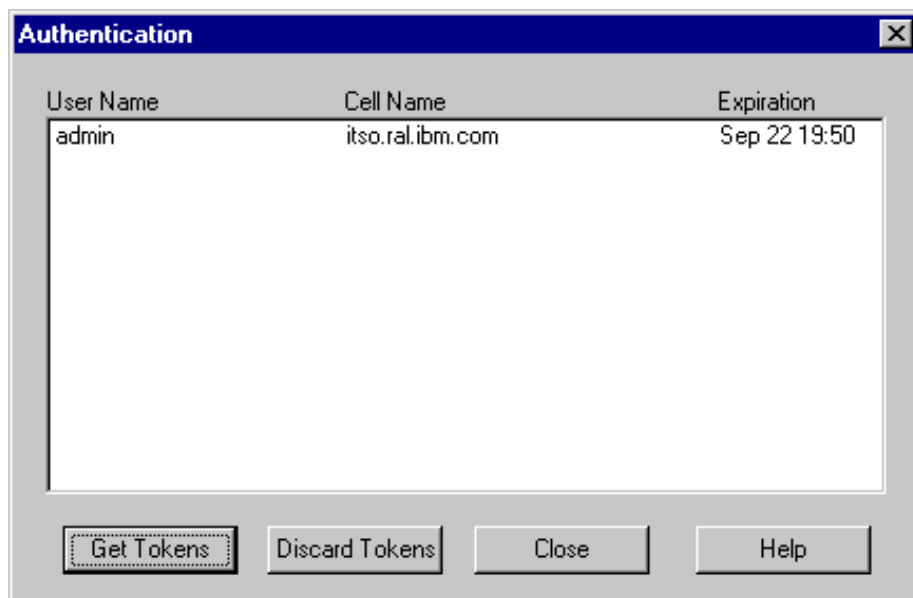


Figure 62. Authentication Complete

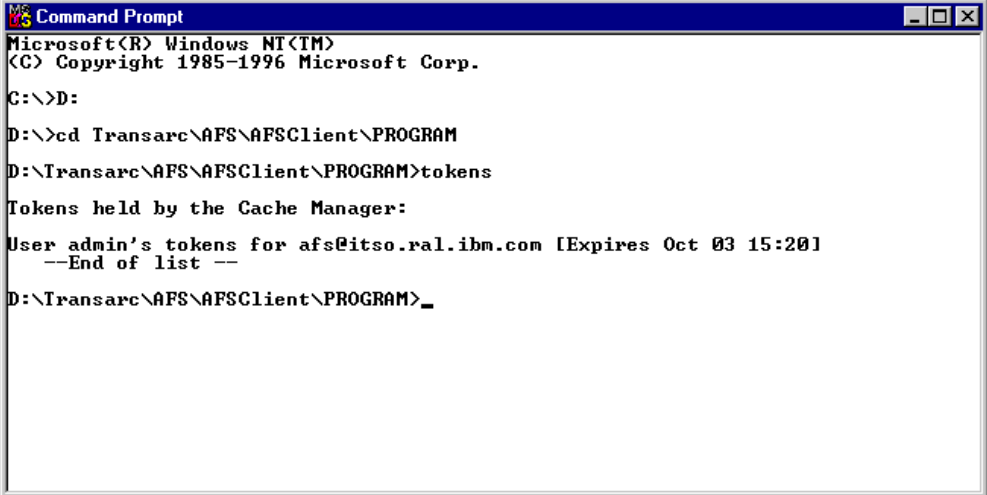
Notice that you could also have entered the following command from a Command Prompt window to authenticate as admin:

```
klog admin
```

The password is not sent in the clear during this authentication process (as we verified using Microsoft Systems Management Server Network Monitor Version 4.00 for Windows NT), but it is encrypted using the Kerberos technology.

Another confirmation that the AFS user admin has been authenticated can be achieved through the `tokens` command (also available on AIX), which must be

launched from a Command Prompt window. (It does not work if you double-click on the icon of the corresponding file tokens.exe.) We first changed the directory to D:\Transarc\AFS\AFSCClient\PROGRAM, where all the AFS executable are stored, and then we entered the `tokens` command. The following figure shows the output we got:



```
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.
C:\>D:
D:\>cd Transarc\AFS\AFSCClient\PROGRAM
D:\Transarc\AFS\AFSCClient\PROGRAM>tokens
Tokens held by the Cache Manager:
User admin's tokens for afs@itso.ral.ibm.com [Expires Oct 03 15:20]
--End of list --
D:\Transarc\AFS\AFSCClient\PROGRAM>_
```

Figure 63. Output of the `tokens` Command

Now that you are authenticated as the AFS user `admin`, you can access the directory `/afs` and its subdirectories `/afs/itso.ral.ibm.com` and `/afs/.itso.ral.ibm.com`, which are in the AFS file system we mounted in 2.4.2, “Configuration of the First AFS Machine on AIX” on page 29. You can also list or edit files if you have been granted the rights to do this.

Another way to authenticate would have been by opening the File Manager through the `winfile` command, open the AFS menu, which has been added by the AFS installation, and then click on **Authentication...**, as shown in the following figure:

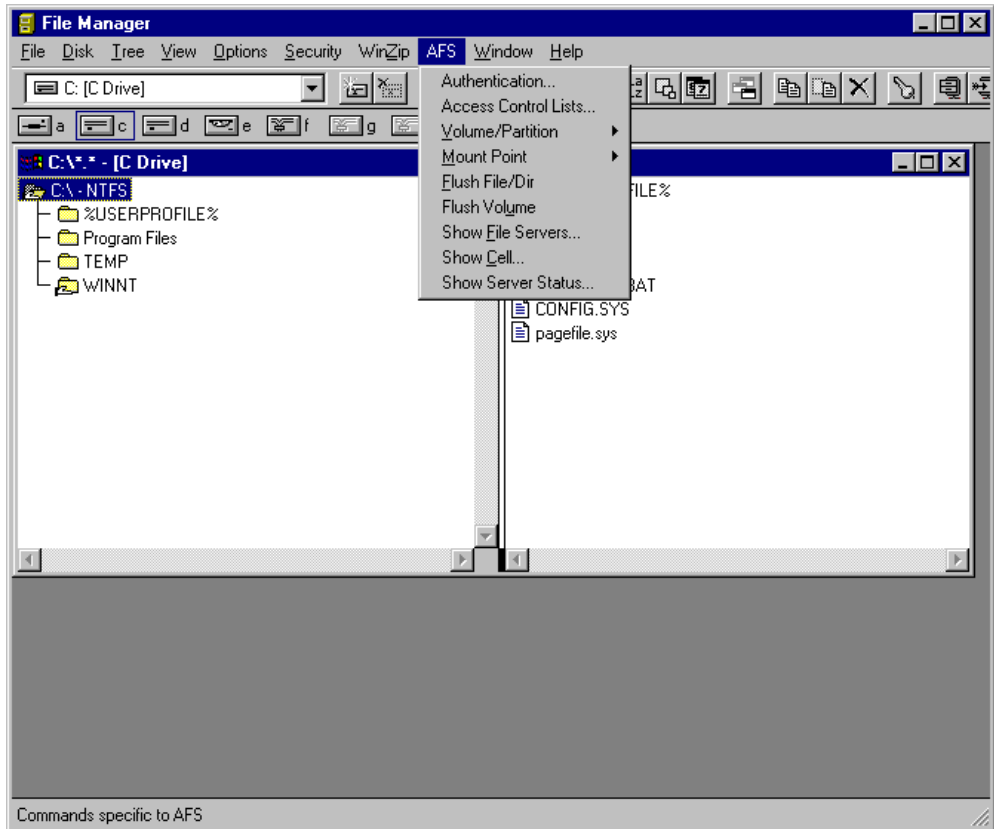


Figure 64. AFS Menu in the File Manager Window

The AFS menu in the File Manager window is added by the AFS client installation for Windows NT. Many commands on the AIX platform that can be launched from the command line have a correspondent GUI version in this menu.

Moreover, in Windows NT Explorer, we right-clicked on **all on 'wtr05195-afs' (H:)**, which was the network drive where the AFS file system had been mounted, and we saw that the same AFS menu had been generated also in the pop-up menu of that network drive, as shown next:

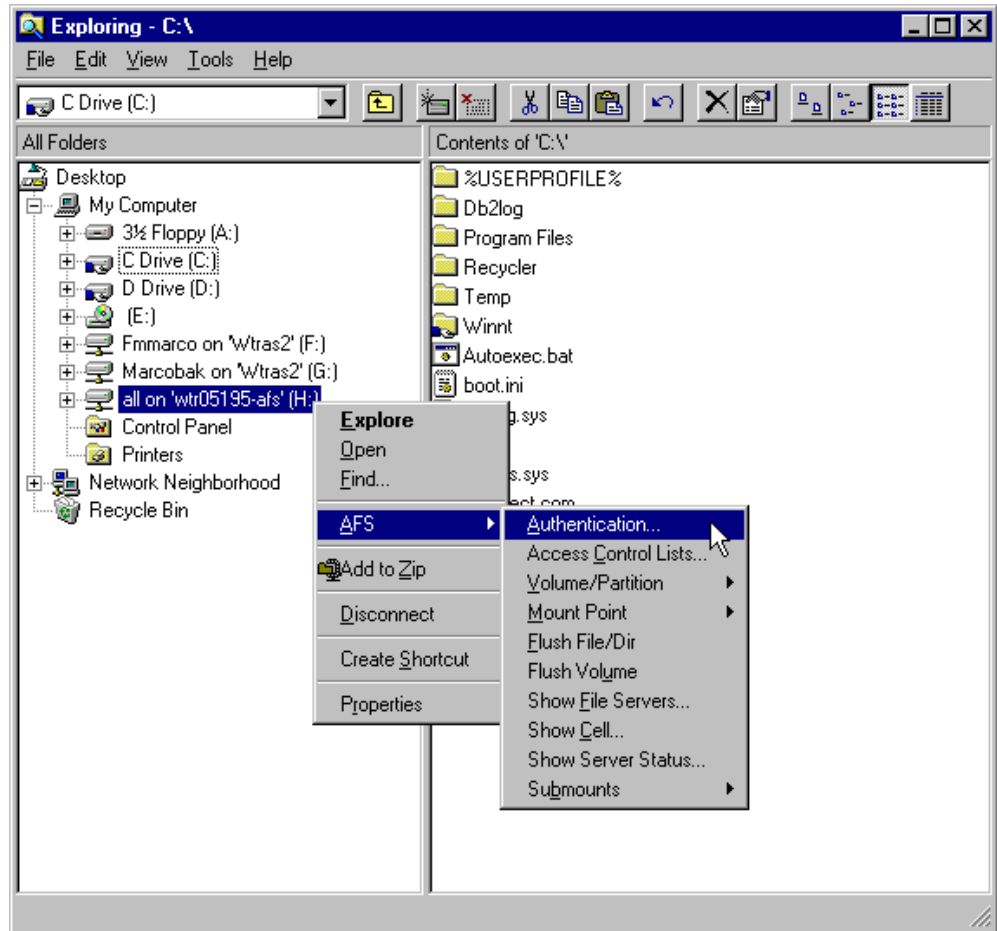


Figure 65. AFS Pop-Up Menu

This option offers another possibility to authenticate.

2.5 Creation of an AFS User

We show you now the steps to create an AFS user. We created an AFS user on our first AFS machine (see 2.4.1, "Installation of the First AFS Machine on AIX" on page 22 and 2.4.2, "Configuration of the First AFS Machine on AIX" on page 29). Actually you could also issue the various commands necessary to create an AFS account from a login session at an AIX or Solaris AFS client machine. The information about the user's account (database entries, volume information, etc.) is actually stored on the File Server and Database Server Machines.

First, it is important to clarify that an AFS account and a UNIX or Windows NT account are not the same thing. As we will see, you can authenticate to AFS without being defined as a user on the operating system of any of the machines in the cell, and you could be a user on the operating system of some machines in the cell without being a recognized AFS user.

To define an AFS account, you must follow these steps:

1. Create an entry in the Protection Database:

This entry maps the user name to the AFS user ID. You will refer to AFS users through their user names, but the AFS server will refer to them by their AFS user IDs. The Protection Database entry also records all the protection groups to which the user belongs.

2. Create an entry in the Authentication Database.

This entry records the user's password in a scrambled form suitable for use as an encryption key.

3. Create the AFS user's volume.

The user's volume is like a logical container that keeps all the files of a single user in the AFS user's home directory together on a single partition of a File Server Machine.

4. Assign the user's volume a quota.

The space quota is the maximum size that the user's volume can consume. By default this value is set to 5MB, but you can alter it if you wish.

5. Define a mount point that associates the AFS user's volume with the AFS user's home directory.

For example, in our AFS server configuration (see 2.4.2, "Configuration of the First AFS Machine on AIX" on page 29), the mount point made the contents of the volume visible and accessible in the file system tree `/afs/itso.ral.ibm.com`.

6. Give full ownership and access rights to the AFS user on the home directory.

In this way the AFS user is able to create subdirectories, create, edit and delete files, and so on. In standard UNIX sense, a user must own his or her own home directory. In the AFS sense, a user must have all the available access rights.

Now we describe to you, step by step, the commands we entered to create an AFS account. Creating AFS accounts is an operation that must be performed on the AFS server machine.

2.5.1 Authenticating As admin

We were already logged in to our first AFS machine, `rs600030`, as `root`. We entered the following command in order to have AFS administration authority:

```
klog admin
```

We were prompted to enter admin's password. We typed the password and pressed the Enter key.

The first thing we did after logging on as `admin` was to check if `admin` had administrative rights in the directory `/afs/itso.ral.ibm.com` where we were going to mount the user's volume. To do this, we issued the following two commands:

```
cd /usr/afs/bin
./fs listacl /afs
```

The output produced was the following:

```
Access list for /afs is
Normal rights:
  system:administrators rlidwka
  system:anyuser rl
```

This output gave us the confirmation that the user admin, member of system:administrators, had all the possible rights under /afs, while users of the group system:anyuser had only the rights of read and lookup.

Possible AFS Rights and Their Meanings

In order to understand the output of the command `fs listacl directory_name` it would be useful to list the possible AFS rights and their meanings:

- `r` - *read* the contents of files in the directory
- `w` - *write* or modify the contents of files in the directory
- `l` - *lookup* status information about the files in the directory
- `d` - *delete* files from the directory
- `i` - *insert* new files into the directory
- `k` - *lock* - set read or write locks on the files in the directory
- `a` - *administer* - change the rights on the ACL

2.5.2 Creation of an Entry in the Protection Database

Next we created an entry in the Protection Database. The command to do this has the following syntax:

```
pts createuser user_name
```

If you want to assign the user a specific user ID, you can enter:

```
pts createuser user_name user_ID
```

If you do not specify any ID for the user you are creating, the Protection Server normally allocates the next available AFS user ID. We were interested in controlling the user ID that would be assigned to the new AFS user. For this reason we entered:

```
pts createuser afsuser 150
```

This command produced the following confirmation output:

```
user afsuser has id 150
```

We explain in 2.6.1, "AFS Integrated Logon on AIX" on page 91 the reason why we entered 150 as the user ID and not another number.

Notice that if the user ID that you specify is already in use, you would receive an error message and you should specify a different user ID for your command to take effect.

2.5.3 Creation of an Entry in the Authentication Database

We created an entry in the Authentication Database with the command:

```
kas create afsuser -admin admin
```

After issuing the above command, we were prompted to enter admin's password:

```
Administrator's (admin) password:
```

Then we were prompted to enter twice the password for afsuser:

```
initial_password:  
Verifying, please re-enter initial_password:
```

Had we only entered the following we would not have been able to create the new user:

```
kas create afsuser
```

In fact in that case the AFS system would have tried to authenticate us as root, but root is not a recognized AFS user.

2.5.4 User's Volume Creation

Next, we created the user's volume user.afsuser through the following command:

```
vos create rs600030 /vicepa user.afsuser
```

The output was:

```
Volume 536870930 created on partition /vicepa of rs600030
```

It's specifically recommended that the name for the user's volume has the following form:

```
user.user_name
```

2.5.5 User's Volume Mount Point

Now we show you how to mount the volume in the AFS directory. It is recommended, although not mandatory, that you have another level in the directory structure, where users' home directories should be located; otherwise all user directories would be under the cell name and this would create confusion when you want to create and attach non-user volumes. So the path should be */afs/cell/user/username*. However, we were just experimenting to see what steps are required to mount the volume in the AFS directory. Since we would not have created or attached any non-user volumes, we just entered the following command without creating another level in the directory structure:

```
./fs mkmount /afs/.itso.ral.ibm.com/afsuser user.afsuser
```

Notice that the mount point serves as the root directory of the volume, since it associates a directory name with a volume. In our case, the user's volume was user.afsuser, and the user's home directory was /afs/itso.ral.ibm.com/afsuser. You probably noticed that we had not previously issued any commands to create the user's home directory. The reason is that the above command automatically creates the user's home directory and then mounts the user's volume over the home directory.

Another important thing that we want to underline about the above command is that we did not specify /afs/itso.ral.ibm.com/afsuser as the user's home directory,

but /afs.itso.ral.ibm.com/afsuser. In fact you have to make the mount point in the ReadWrite version of a specific volume and release it. In order to release it, you must issue this command:

```
vos release root.cell
```

You should see the following output:

```
Released volume root.cell successfully
```

2.5.6 Full Ownership and Access Rights on the Home Directory

Next we gave all the available access rights to the AFS user afsuser on the home directory /afs.itso.ral.ibm.com/afsuser. To do this, we entered the following command:

```
./fs setacl /afs.itso.ral.ibm.com/afsuser afsuser all
```

Then, to verify that afsuser had been granted all the available access rights, we entered the command:

```
./fs listacl /afs.itso.ral.ibm.com/afsuser
```

The output produced is shown in the following screen:

```
Access list for /afs.itso.ral.ibm.com/afsuser is
Normal rights:
  system:administrators rlidwka
  afsuser rlidwka
```

2.5.7 Setting a Space Quota on the Volume

As we mentioned in 4 on page 86, the default quota for the user's volume as soon as you create it is 5MB. For our scenarios (see Part 2, "IBM WebSphere Performance Pack Scenarios" on page 347), we considered this value insufficient, and we decided to grant an unlimited quota for the user.afsuser volume we had created. For this reason, we issued the following command on the AFS server machine rs600030:

```
./fs setquota /afs.itso.ral.ibm.com/afsuser 0
```

However, as we said, we made this in prevision of the scenarios we would implement, but in general it is not recommended to trust a user with an unlimited quota.

2.5.8 Setting an Appropriate Access Control List

As we have shown, the user afsuser got full access rights to the home directory /afs.itso.ral.ibm.com/afsuser. The only problem is that when you create a new volume and you define a mount point that associates the AFS user's volume with the AFS user's home directory, by default all the users in the system:administrators group also get full access rights, and members of system:anyuser group do not get any rights. We considered it necessary to change this default ACL entry and we gave to all the users in both the system:administrators and system:anyuser groups only the AFS lookup right. This we did by issuing the following two commands:

```
./fs setacl /afs.itso.ral.ibm.com/afsuser system:administrators l
```

```
./fs setacl /afs/itso.ral.ibm.com/afsuser system:anyuser l
```

To verify that our command had been successfully executed, we entered:

```
./fs listacl /afs/itso.ral.ibm.com/afsuser
```

The output confirmed that members of the system:administrators and system:anyuser groups have only lookup permission, while the user afsuser has all the seven possible rights:

```
Access list for /afs/itso.ral.ibm.com/afsuser is
Normal rights:
  system:administrators l
  system:anyuser l
  afsuser rlidwka
```

2.5.9 AFS User Authentication

It is now possible to authenticate to AFS as afsuser from an AFS client machine. To do this, it is necessary to log on into that machine as an already defined operating system user, such as root on AIX or Administrator on Windows NT. In other words, a user that wants to authenticate to AFS from an AFS client machine should have an operating system account. We will see in Figure 2.6 on page 91 how this procedure can be optimized.

The steps to authenticate to AFS as afsuser from an AFS client machine on AIX are similar to the procedure described in 2.5.1, “Authenticating As admin” on page 86. The only difference is that in this case the command would be:

```
klog afsuser
```

The authentication from an AFS client machine running on Windows NT is similar to the steps shown in 2.4.7.5, “Authenticating As admin” on page 81, the only difference being that you should enter afsuser’s user name and password in the Get Tokens panel shown in Figure 61 on page 82.

2.5.10 Experimenting with Access Control Lists

We wanted to try an experiment with the access control list we had defined on the afsuser home directory. We authenticated as afsuser and created a text file pippo.htm with the vi editor. This text file was saved in the afsuser home directory. Then we authenticated again, this time as admin, and we accessed the afsuser home directory by entering:

```
cd /afs/itso.ral.ibm.com/afsuser
```

This operation completed successfully, and we could even list the contents of this directory, since user admin had been granted lookup rights. However, when we tried to browse the contents of that text file by issuing the following command it did not give any output:

```
more pippo.htm
```

We tried then to open the file pippo.htm with the vi editor by issuing:

```
vi pippo.htm
```


The system did not permit us to do this operation, and the following error message was displayed:

The file access permissions do not allow the specified action

This confirmed that the ACL on the afsuser's home directory had been set to only allow lookup permission.

A similar error message also appeared on the Windows NT AFS client machine when we tried to open the same text file with WordPad, after authenticating as admin, as shown in the following screen:

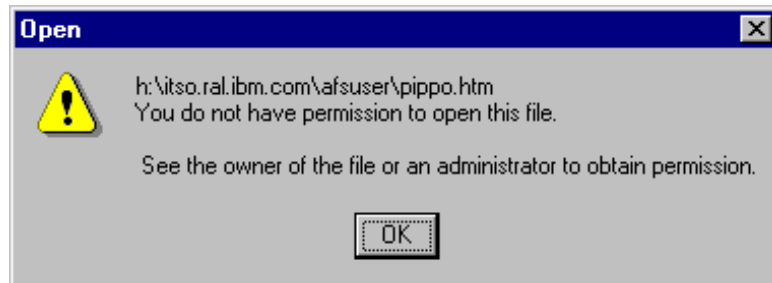


Figure 66. Error Message Displayed When Trying to Access a Protected File

2.6 AFS Integrated Logon

The *AFS Integrated Logon* on a specific platform is the process that permits an AFS user to log in to an AFS client machine as a user being recognized in one single step by the operating system of the AFS client machine and by the AFS file system.

If you do not follow this procedure, each AFS user will have to log in twice: the first time on the operating system of the AFS client machine, and the second time on the AFS file system.

2.6.1 AFS Integrated Logon on AIX

Our goal of this section is to explain how to enable the Integrated Logon on AIX. With the Integrated Logon on AIX, an AFS user will log in with the AFS login, which is a modified version of the AIX `login` command. This grants the user access to the local operating system and authenticates the user to AFS.

2.6.1.1 Creation of an AIX User

First, it is necessary that an AFS user has an entry in the local password file `/etc/passwd` of each AFS client AIX machine he or she will log onto. In other words, an AFS user must be defined as an AIX user in all the AFS client machines the user will access to log on.

Each user has an AIX user ID (a unique decimal integer string to associate with this user account on the system) and an AFS user ID (a unique decimal integer string that uniquely identifies the user in the AFS cell). The AIX user ID is defined in the `/etc/passwd` file of each AIX workstation from which the user will log in. The AFS user ID is defined in the Protection Database of the AFS server machine. For each user that will be an AIX and AFS user, it is logical to define the AFS user

name equal to the AIX user name. To implement the AFS Integrated Logon, it is also necessary that the user's AIX and AFS user IDs are the same.

We defined an AIX user, named afsuser on each AIX workstation the user would log into. To do this, we used the smitty application. We logged in as root on each AFS client machine, and then we entered:

```
smitty users
```

In the smitty screen, we selected **Add a User** and then we defined the afsuser user. The following screen, which we captured on our First AFS server machine rs600030, summarizes the particulars for the new user created:

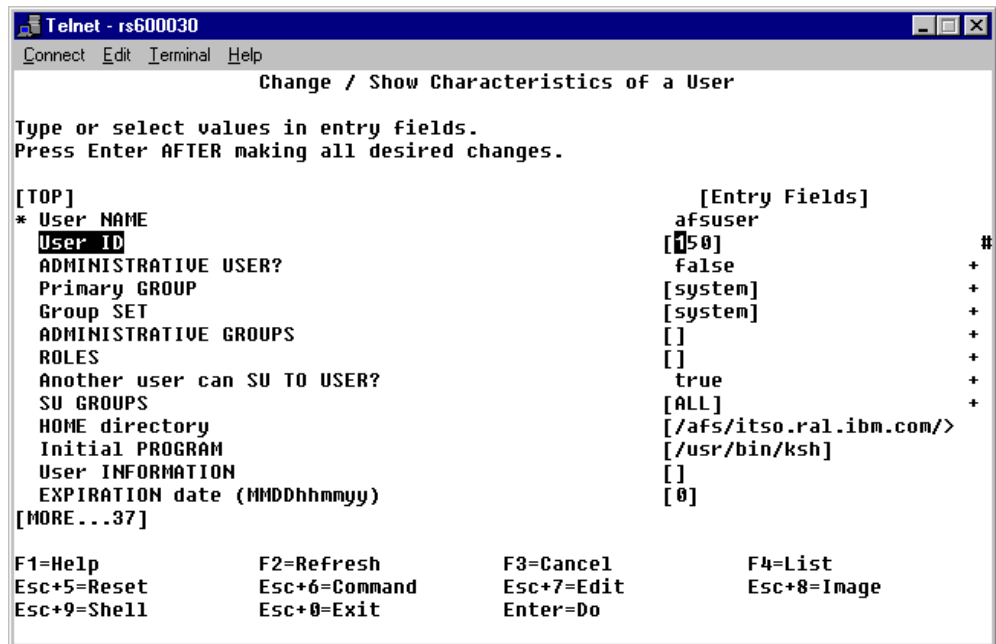


Figure 67. Particulars of the afsuser UNIX User

At this moment, afsuser is still a normal AIX user. However, notice that we did not let the operating system decide what user ID would be assigned to afsuser. When you define a new user, if you leave the User ID field blank, it's the operating system itself that decides that value, choosing from the pool of all the values that have not yet been assigned to any other users. Instead, we explicitly entered the value 150 after checking that such a number had not been assigned to any AIX users in all the other AFS client AIX machines. As we have already mentioned, all the assigned AIX user IDs are visible in the /etc/passwd file. Notice that 150 was also the value we entered as the AFS user ID for afsuser in 2.5, "Creation of an AFS User" on page 85.

Next, we exited from smitty and added an entry for afsuser in the file /etc/security/passwd by issuing the following command and entering a password for afsuser when the system prompted us:

```
passwd afsuser
```

This we did in all the AFS client machines that ran the AIX operating system.

2.6.1.2 Home Directory Ownership

It is also necessary that an AFS user owns his or her home directory on the UNIX system on which it had been defined. To perform this operation, we logged in as root on the AFS server machine rs600030, by entering:

```
login root
```

We had to type root's password before effectively logging in. Then we entered the following command, where 150 was the afsuser's user ID on AFS and AIX (see Figure 67 on page 92):

```
chown 150 /afs/itso.ral.ibm.com/afsuser
```

To verify that afsuser was the owner of the /afs/itso.ral.ibm.com/afsuser directory, we entered the following command on the AFS client machine rs600012:

```
ls -l /afs/itso.ral.ibm.com
```

The output we got for the afsuser subdirectory gave us the confirmation we needed:

```
drwxrwxrwx  2 afsuser  system      4096 Sep 19 20:51 afsuser
```

2.6.1.3 Enabling AFS Login

First, make sure that the file afs_dynamic_auth exists in every AFS client machine from which the user will log in. Such a file must be located in /usr/vice/etc and must also be executable.

After the original installation of the File Sharing client component of IBM WebSphere Performance Pack, we found this file in our AFS client AIX machines under the directory /public/WebSphere/AFS/usr/afsws/root.client/usr/vice/etc. Then we copied it under /usr/vice/etc with the command:

```
cp /public/WebSphere/AFS/usr/afsws/root.client/usr/vice/etc/afs_dynamic_auth /usr/vice/etc
```

We made it executable by entering:

```
chmod 744 /usr/vice/etc/afs_dynamic_auth
```

Next, you have to modify the files /etc/security/user and /etc/security/login.cfg. Before making these modifications, we recommend that you save copies of the original files with different names. For example, we entered the following two commands:

```
cp /etc/security/user /etc/security/user.original  
cp /etc/security/login.cfg /etc/security/login.cfg.original
```

In the following figure, we show you the two sections, default and root, that we modified in the /etc/security/user file. The modifications we made are highlighted in boldface text:

```

default:
    admin = false
    login = true
    su = true
    daemon = true
    rlogin = true
    sugroups = ALL
    admgroups =
    ttys = ALL
    auth1 = SYSTEM
    auth2 = NONE
    tpath = nosak
    umask = 022
    expires = 0
    SYSTEM = "AFS OR (AFS[UNAVAIL] OR compat[SUCCESS])"
    registry = DCE
    logintimes =
    pldwarntime = 0
    account_locked = false
    loginretries = 0
    histexpire = 0
    histsize = 0
    minage = 0
    maxage = 0
    maxexpired = -1
    minother = 0
    minlen = 0
    mindiff = 0
    maxrepeats = 8
    dictionlist =
    pwdchecks =

root:
    registry = files
    admin = true
    SYSTEM = "compat"
    loginretries = 0
    account_locked = false

```

Note

If you follow the directions to enable the AFS login on AIX found in the guide *AFS Installation Guide* (shipped with the product, but not separately orderable), you will see there a small difference that what we have described here. The difference is that, for AIX machines that are only AFS clients, they recommend you to add the following line in the `default` section of the `/etc/security/user` file:

```
SYSTEM = "AFS OR AFS [UNAVAIL] AND compat [SUCCESS]"
```

The problem is that this modification does not work.

In the text file `rsaix42install.txt`, which we found in the directory `/public/WebSphere/doc/enu/transarc`, we read that the correct line to add for AIX 4.2.1 and AIX 4.3 should be:

```
SYSTEM = "AFS OR (AFS[UNAVAIL] OR compat[SUCCESS])"
```

Although this appears to be a small difference, it is necessary to properly enable AFS Integrated Logon.

Notice that if the machine is both an AFS and a DCE client, the `SYSTEM` variable in the `default` section of the `/etc/security/user` file should be set as follows:

```
SYSTEM = "DCE or DCE [UNAVAIL] OR AFS OR AFS [UNAVAIL] AND compat[SUCCESS]"
```

In the file `/etc/security/login.cfg`, we added the following at the bottom of the file:

```
DCE:
program = /usr/vice/etc/afs_dynamic_auth

AFS:
program = /usr/vice/etc/afs_dynamic_auth
retry = 3
timeout = 30
retry_delay = 10
```

When the AFS user enters the AIX command `login`, the system does not execute the standard login program, but the AFS authentication program `afs_dynamic_auth`, which leads the user to log in on AIX and AFS simultaneously.

Notice that it is not necessary that the AIX and AFS passwords for a user be the same when the AFS Integrated Logon on AIX is activated. In fact, it is the `afs_dynamic_auth` authentication program that takes the precedence in terms of authentication. As long as the user provides the correct AFS password, that user is authenticated directly to both AFS and AIX, and the authentication program `afs_dynamic_auth` does not even consult the AIX password file `/etc/security/passwd`. We see more details on this in 2.6.2, "Interactions between the AFS Login and the AIX Login" on page 97.

2.6.1.4 Modifying the /etc/passwd File

When we opened the /etc/passwd file with the vi editor, the entry for the afsuser user was:

```
afsuser:*:150:1::/home/afsuser:/usr/bin/ksh
```

We wanted to change one parameter in the above entry to place the user in his or her AFS home directory. In fact, it is more logical that an AFS user logs in directly in his or her AFS home directory, rather than the AIX home directory. For this reason, we replaced /home/afsuser with /afs/itso.ral.ibm.com/afsuser, and the above entry in the /etc/security/passwd file became:

```
afsuser:*:150:1::/afs/itso.ral.ibm.com/afsuser:/usr/bin/ksh
```

We executed the above modification in all the AFS client machines on AIX.

2.6.1.5 Experimenting with the AFS Integrated Logon on AIX

To experiment with the AFS Integrated Logon on AIX, make sure that you destroy your current tokens on your AFS client machine. To do this, on our AFS client machine rs600012, we entered the command unlog. The following command would have worked as well, but we recommend unlog since kas forgetticket is normally limited to system administrators:

```
/afs/usr/bin/kas forgetticket rs600012
```

Then, check to make sure that the above command took effect and destroyed your current tokens. To do this, enter the following command:

```
/usr/afs/bin/tokens
```

This is the output you should see:

```
Tokens held by the Cache Manager:  
  
--End of list--
```

After our tokens were destroyed, we were ready to test if our configuration of the AFS Integrated Logon on AIX was successful. To do this, we logged in as afsuser on our AFS client machine rs600012, by issuing the login command and entering afsuser as the user name. The corresponding output was:

```
AIX Version 4  
(C) Copyrights by IBM and by others 1982, 1996.  
login: afsuser
```

Then we entered the password and logged into the AFS client machine. To verify that afsuser was placed in the AFS home directory, we issued the pwd command, and we got the following output, which confirmed that our modification to the file /etc/security/passwd had taken effect:

```
/afs/itso.ral.ibm.com/afsuser
```

We also wanted to verify that we had been successfully authenticated to the AFS file system. To do this, we again entered the command:

```
/usr/afs/bin/tokens
```

This time the output we got was:

```
Tokens held by the Cache Manager:

User's (AFS ID 150) tokens for afs@itso.ral.ibm.com [Expires Sep 23 18:06]
--End of list--
```

We had the confirmation that the Cache Manager of our AFS client machine was holding tokens for the AFS user with user ID 150, which was afsuser.

This completes the demonstration of how it is possible to enable the AFS Integrated Logon on the AIX platform.

2.6.2 Interactions between the AFS Login and the AIX Login

As we have shown in 2.6.1, “AFS Integrated Logon on AIX” on page 91, when you complete the procedure we described to enable AFS Integrated Logon, an AFS user logging in is automatically authenticated to AFS. As long as the user provides the correct AFS password, the AFS login never consults the local password file.

In this situation, root is the only non-AFS user that can log into the AIX machine without being authenticated to the AFS file system. All the other users must pass through the AFS login, so only authorized AFS users can log into the AFS file system and the AIX machine.

We want to show you now how you can configure your AIX system to accept other AIX users, besides root, that are not AFS users.

On our AFS client AIX machine, we created through smitty a new AIX user named madtest. To do this, we simply entered, as root, the following command:

```
smitty users
```

Then we selected **Add a User**. In the window that was displayed, we entered madtest as the name of the new user, and left all the other fields to their default values. As soon as you press the Enter key, madtest is a new AIX user added to your system and you will find a new entry for madtest in the file /etc/passwd. However, this user still falls under the default user category, for which the AFS login takes precedence on the AIX login. You cannot even set a password for madtest. We tried with both smitty and the passwd command, but no entry was added to the /etc/security/passwd file. For this reason, you cannot log in as madtest yet, since no password has been defined for this user.

After this step, we opened the file /etc/security/user with the vi editor, by entering:

```
vi /etc/security/user
```

Then we modified the entry for madtest. We added the two lines that are highlighted in boldfaced test in the following screen:

```
madtest:
  admin = false
  SYSTEM = "compat"
  registry = files
```

This way the new user madtest does not fall under the default user category, for which the AFS login takes the precedence. You can see that we have modified the entry for madtest so that now it has become similar to the entry for root.

We issued again, as root, the command:

```
passwd madtest
```

This time we were able to set a password for madtest and to see this password scrambled in the `/etc/security/passwd` file. After this step, we successfully logged in as madtest. Notice that when we logged in as madtest, it was the AIX login that authenticated us, rather than the AFS login. An AFS entry named madtest does not even exist. To be sure that the AFS login had not come into the scene, we entered the command:

```
/usr/afs/bin/tokens
```

We saw the following output:

```
Tokens held by the Cache Manager:

--End of list--
```

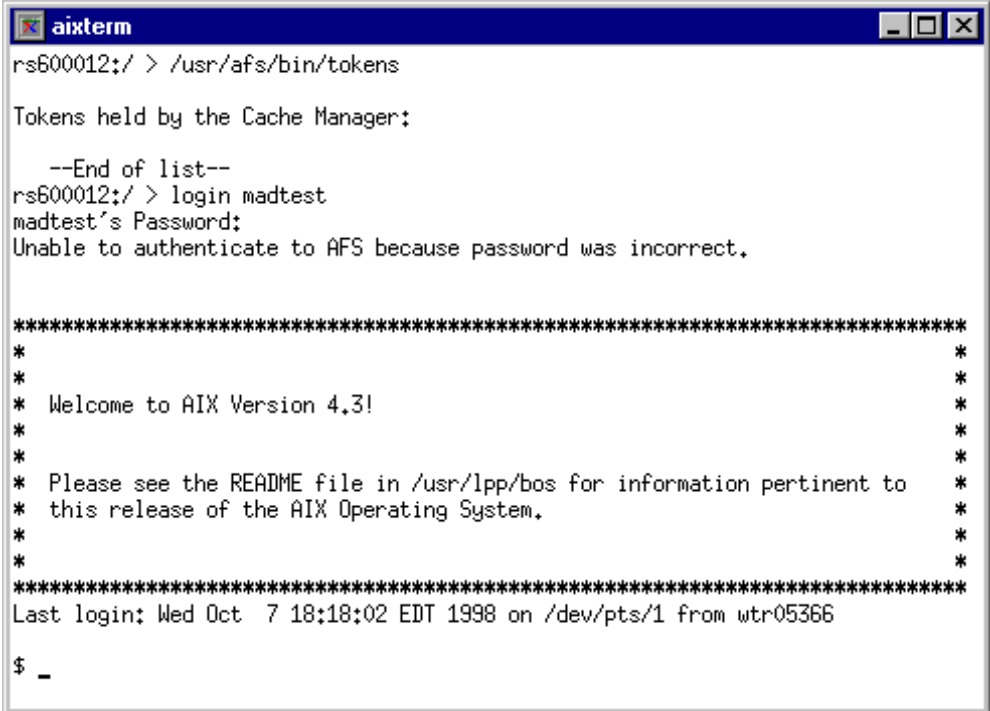
We want to show you now how you can use the local password file `/etc/security/passwd` to still control access even with AFS users:

1. Entering an asterisk `*` in the password field prevents the user from either authenticating with AFS or logging into the machine's AIX file system. This is very useful in case of an emergency, if you want to prevent certain users from logging into the machine.
2. Entering in the password field a string of any length other than the standard thirteen characters (you usually entered a simple `x`) means that if the user does not provide the correct AFS password, they will also be not allowed to log into the local machine's AIX file system. This is appropriate if you want to make sure that only people with AFS accounts in your cell can log into your machines.
3. Putting an actual scrambled password in the password field by using the `passwd` command means that if the user does not provide the correct AFS password, he or she can still log in to the local AIX file system by providing the correct AIX password. In this case the AFS and AIX passwords should differ, or a non correct AFS password would prevent the user from authenticating into the AIX system as well.

We successfully experimented with all the three cases above. Notice that in 3 on page 98, the AFS and AIX passwords should differ, or an incorrect AFS password would prevent the user from authenticating into the AIX system as well.

We edited again the file `/etc/security/user`, to remove the modification we made to the `madtest` entry. In this way, `madtest` fell again into the default user category, for which the AFS login takes precedence. Then we defined a new AFS account, still named `madtest`. To do this we followed the same procedure that we indicated in 2.5, "Creation of an AFS User" on page 85. In particular, the AFS and AIX user IDs for `madtest` were the same, but the passwords were different. We tried to log in as `madtest` by entering the correct AFS password and the AFS login authenticated us on both AFS and AIX, without checking the entry for `madtest` in the `etc/security/passwd` file. We also saw that a token had been produced on the AFS client machine `rs600012` for `madtest`.

We then discarded that token, and logged out. When we logged in again as `madtest`, this time we entered the password for the AIX authentication. This time the AFS login program did not authenticate us, since it found an incorrect password, but since we had entered the correct AIX password at least we were able to access the AIX machine as a normal AIX user, as shown in the following screen:



```
rs600012:/ > /usr/afs/bin/tokens
Tokens held by the Cache Manager:
--End of list--
rs600012:/ > login madtest
madtest's Password:
Unable to authenticate to AFS because password was incorrect.

*****
*
* Welcome to AIX Version 4.3!
*
*
* Please see the README file in /usr/lpp/bos for information pertinent to
* this release of the AIX Operating System.
*
*
*****
Last login: Wed Oct 7 18:18:02 EDT 1998 on /dev/pts/1 from wtr05366
$ _
```

Figure 68. Authenticating on AIX but Not on AFS

As you can see from the above figure, before logging in, no tokens were held by the Cache Manager. On entering the correct AIX password, we were not able to authenticate to AFS, since that password was considered incorrect for AFS. The following error message appeared:

```
Unable to authenticate to AFS because password was incorrect
```

However, after that, the password was compared with the scrambled password stored in the `/etc/security/password` file, and we were able to authenticate to the AIX system.

2.6.3 AFS Integrated Logon on Windows NT

In this section, we show you the steps to enable the AFS Integrated Logon on the Windows NT platform. To know more details about the AFS client Windows NT machine that we used, refer to 2.4.5, “Installation of a Stand-Alone AFS Client on Windows NT” on page 58.

First, we opened the AFS Client Configuration panel. To do this, it is necessary to double-click on the **AFS Client** icon that is automatically created in the Control Panel during the installation of the File Sharing client component of IBM WebSphere Performance Pack. The following figure shows the window that was brought up:

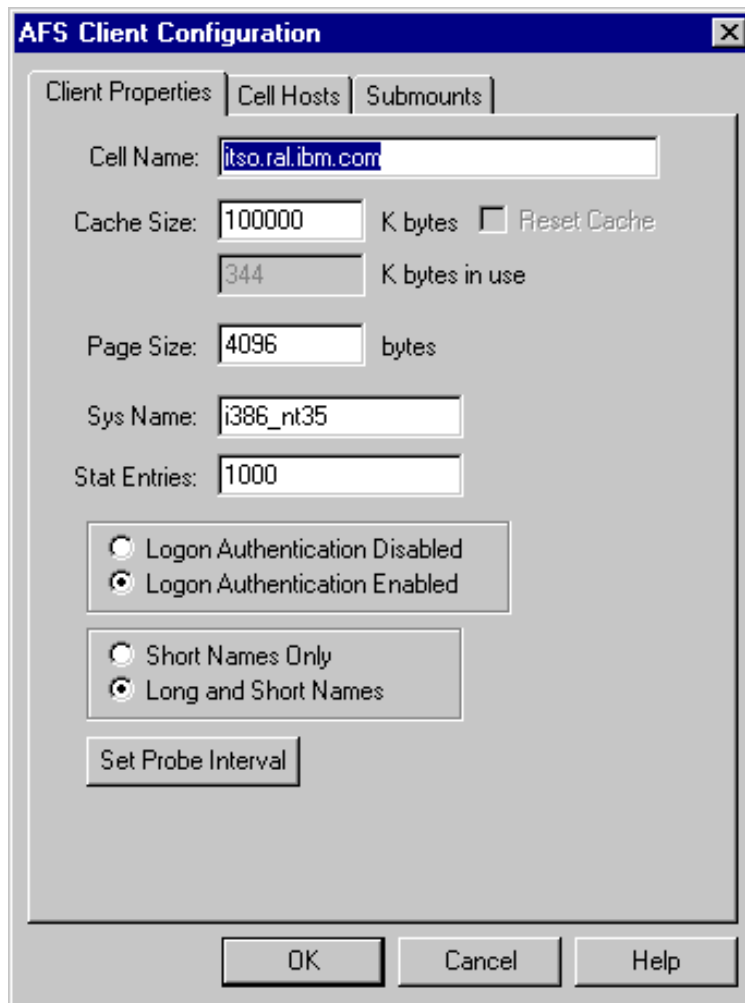


Figure 69. AFS Configuration Client Panel

If you compare the above figure with Figure 42 on page 66 (which also represented the AFS Client Configuration panel displayed during the initial configuration of the AFS client) you will see that now you have the possibility to enable or disable Logon Authentication. We checked the radio button **Logon Authentication Enabled** in order to allow users to authenticate themselves on Windows NT and AFS in one single step. This operation is known as AFS Integrated Logon, and we have already seen in 2.6.1, “AFS Integrated Logon on AIX” on page 91 how you can get it on the AIX platform.

In order to permit the Integrated Logon, after checking the **Logon Authentication Enabled** radio button in Figure 67 on page 92, you should create a new Windows NT account that *must* have the same user name and password as an existing AFS account. In 2.5, “Creation of an AFS User” on page 85, we had created an AFS user named afsuser on our first AFS machine rs600030. So we could create a new account on Windows NT having the same user ID and password as the AFS user afsuser, as shown in the following figure:

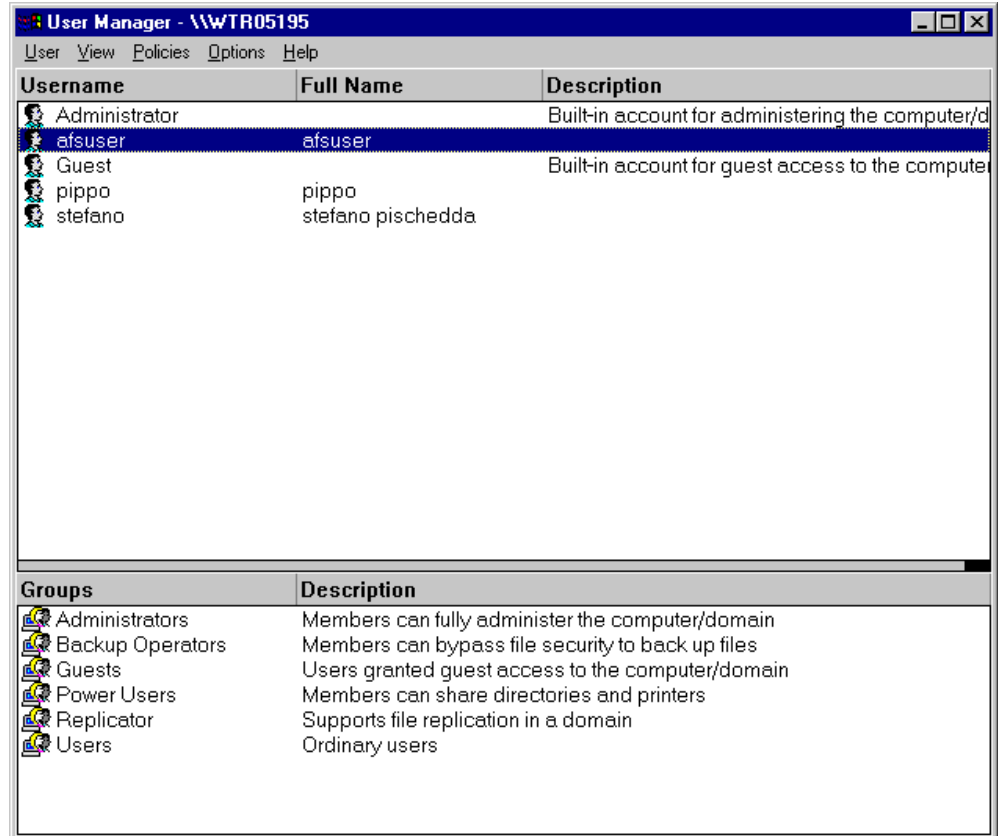


Figure 70. Creation of the Windows NT User afsuser

Notice that it is not necessary for the new user to be a Windows NT administrator; the user afsuser we just created was part of the Users group. It is logical, in fact, that new users that must pass through a Windows NT workstation to access the AFS file system are nothing more than ordinary members of the Users group, and not administrators of the Windows NT machine. As we have already mentioned, the user afsuser had to have the same password on Windows NT and AFS.

We rebooted the machine and tried to log on Windows NT directly as afsuser, since we wanted to experiment with the Integrated Logon on AFS too. But as soon as we entered the user ID and password for afsuser, the following error message was brought up:

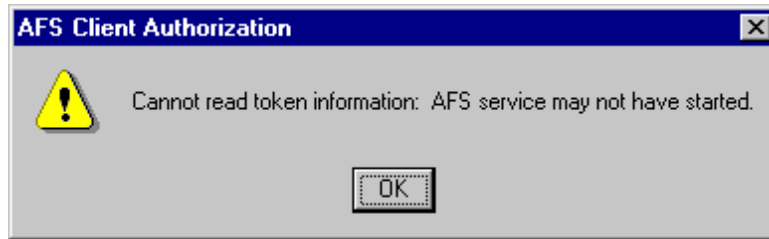


Figure 71. Error Message If the AFS Daemon Has Not Started

The reason of this message is that the Transarc AFS daemon process had not started yet, since we had set the Startup mode to Manual rather than Automatic (see Figure 54 on page 74). We could not experiment with the Integrated Logon yet, because the AFS daemon had not started, and we could log on as afsuser only on Windows NT. Then we tried to start the Transarc AFS Daemon process through the Services panel, clicking on the **Start** button shown in Figure 54 on page 74, but a further error message appeared, since afsuser did not have administrative authority on Windows NT, and could not start that process. We had to log off, close all programs and log into Windows NT as a system administrator. Then we were able to start the AFS daemon.

After this step, we had to log off again, close all programs and log on Windows NT as afsuser. Only at that point were we able to experiment with the Integrated Logon. In order to see the authentication status, we launched the command `afs_auth` from the directory `D:\Transarc\AFS\AFSCClient\PROGRAM` and we got the following screen:

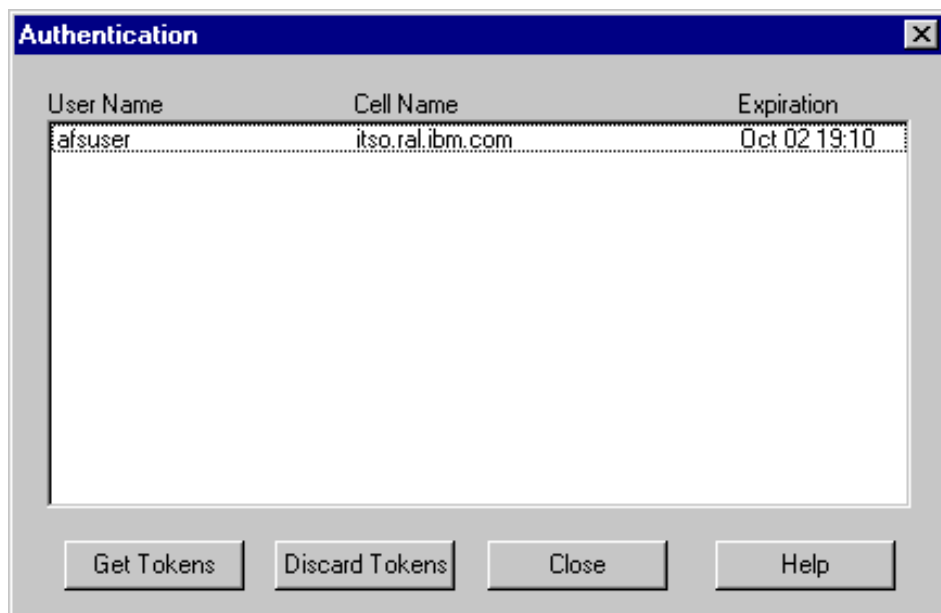


Figure 72. Authentication Panel after Reboot

This window confirmed that the Integrated Logon had worked correctly. In fact we already had the authentication without the need to click the **Get Tokens** button and then enter the user name and password, as shown in Figure 61 on page 82. It had been necessary, however, to log off as afsuser, log on as a system

administrator in order to start the AFS daemon, and then log on as an AFS user again. If you have configured the AFS daemon to start in manual mode, rather than automatic, the AFS Integrated Logon is useful only if you switch from a system administrator user, who can launch the AFS daemon, to an AFS user, as we have shown. But this process is not very useful if you fully shut down your computer and then log into directly as an AFS user. Therefore we recommend that you configure the AFS daemon to automatically start at each system reboot.

If you are configuring your Windows NT AFS client to perform the AFS Integrated Logon, it is also a good idea to set the AFS file system to be mounted at the system startup, as suggested in Figure 56 on page 76.

2.7 A Closer Look at AFS Cached Data Validation

Normal client browser cache recently accessed files locally but did not know whether its copy was fresh or needed to be renewed. The browser could contact the origin server to determine if its copy is still good, but this required more network traffic and delay. AFS uses a proactive approach, since the AFS File Server Machine notifies that there is a new version to all potentially interested clients.

When an AFS user, from an AFS client machine, requests a file to the AFS File Server Machine, the Cache Manager first checks whether that file is already present in the local cache in the AFS client machine. This is true if that file has already been requested. If the file is not there, then the Cache Manager requests it to the AFS File Server Machine, gets the file, stores it in its local cache and serves the file to the application that has requested it (for example, a text editor).

The AFS File Server Machine keeps a list of all the AFS client machines that have requested and cached data associated with a certain file in their AFS cache (either RAM or disk). When that particular file is changed by a user or process that is entitled to do so, the AFS File Server Machine directly contacts all the AFS clients that requested that file in the past, informing them that newer data is now available. This process is also known as *callback*.

The AFS client's kernel can safely return cached data to the application until it is notified by callback. When callback is received, the client must retrieve a fresh copy of the data since the cache is out of data. If an AFS client receives callback from the AFS server, it marks the cached data as invalid. The client will retrieve a fresh copy of the file only when the client really needs it.

This particular distributed approach places a lot of responsibility on the client but speeds up the overall user's response. Most of the data is read from the local disk unless the original file is changed. Network traffic is reduced since local caching requires fewer file accesses and new copies of cached files are requested only if needed.

We also want to point out that *last save wins*. This means that if two users have opened the same file, the first user saves changes, then the second user saves the changes as well, then the second user overwrites changes made by the first user. AFS is not a revision control system.

In this section we describe to you an experience we did on the AFS environment we had set up, in order to understand how AFS really works. We wanted to study

the exchange of information between the AFS File Server Machine and the AFS client machines when a file that the client requests has been modified. To do this we used the `tcpdump` command, launched on our first AFS machine, which was the RISC/6000 rs600030.

First, we want to show you the particulars of the AFS environment we used in this experience:

Table 1. The Particulars of Our AFS Environment

Workstation	First AFS Machine	AFS Client	AFS Client
Operating System	AIX 4.3	AIX 4.3	Windows NT 4.0
IP address	9.24.104.97	9.24.104.124	9.24.104.223
Hostname	rs600030	rs600012	wtr05195

We logged in as the AFS user `afsuser` into the Windows NT AFS client machine `wtr05195`. We have shown in 2.5, "Creation of an AFS User" on page 85, how we created the AFS account `afsuser`, and in 2.6.3, "AFS Integrated Logon on Windows NT" on page 100, you can see how you can configure your Windows NT AFS client to enable the AFS Integrated Logon. The AFS user `afsuser` was the owner of the home directory `/afs/itso.ral.ibm.com/afsuser`.

We list now the ports that are involved in this exchange of information, as they appear in the file `/etc/services`:

- port 7000
This port is used by the file server process `fs` on the AFS File Server Machine (see 9 on page 38).
- port 7001
This port is used on the AFS client machines for AFS callback.
- port 7003
This port is used on the AFS server machine by the Volume Location Server process `vlserver` (see 7 on page 38).

We followed this sequence of operations:

1. From the Windows NT AFS client machine `wtr05195`, we opened a text file located in `/afs/itso.ral.ibm.com/afsuser`
2. We launched the `tcpdump` command on the AFS server, to monitor the traffic generated by the `wtr05195` machine and received on the token-ring network interface `tr0` of the AFS server:

```
tcpdump -I -nt -i tr0 src 9.24.104.223
```

3. We modified and saved the text file we had opened.

This is the output we got as soon as we saved the file:

```
9.24.104.223.7001 > 9.24.104.97.7000: udp 44
9.24.104.223.7001 > 9.24.104.97.7000: udp 636
9.24.104.223.7001 > 9.24.104.97.7000: udp 58
9.24.104.223.7001 > 9.24.104.97.7000: udp 456
9.24.104.223.7001 > 9.24.104.97.7000: udp 57
9.24.104.223.7001 > 9.24.104.97.7000: udp 253
9.24.104.223.7001 > 9.24.104.97.7000: udp 57
```

You can see from the above screen that the wtr05195 machine, from its port 7001, informs the AFS server listening on the port 7000, that the text file has been modified. This is the first step of the callback process.

Of course, the AFS server immediately informs all the AFS clients that in the past had requested that particular file that the next time they will need that file they should retrieve a fresh copy of it, since the cached copy is out-of-date. We wanted to monitor this exchange of information too.

So, first we opened the file from the other AFS client machine that was part of our environment, rs600012, to ensure that such a file was stored in the cache of this client too. Then we repeated the same sequence of operations just described, but this time we issued the `tcpdump` command with different parameters, since we wanted to study all the network traffic exchanged between the three machines, and involving the port 7001:

```
tcpdump -I -nt -i tr0 port 7001
```

We show you the output we got divided in two parts. The first part demonstrates once again that the Windows NT AFS client wtr05195 informs the AFS server that the file has been modified. This exchange of information is generated by the AFS client machine through a callback:

```
9.24.104.223.7001 > 9.24.104.97.7000: udp 291
9.24.104.97.7000 > 9.24.104.223.7001: udp 136
9.24.104.97.7000 > 9.24.104.223.7001: udp 136
9.24.104.223.7001 > 9.24.104.97.7000: udp 57
```

The second part of the output shows how the AFS server also informs the other AFS client rs600012 that the file it had retrieved and cached is now out-of-date:

```
9.24.104.97.7000 > 9.24.104.124.7001: udp 36
9.24.104.97.7000 > 9.24.104.124.7001: udp 36
9.24.104.124.7001 > 9.24.104.97.7000: udp 61
```

Through the above callback, the Cache Manager on the AFS client machine rs600012 has been made aware that the file it retrieved and cached in the past has now been changed, and it will have to retrieve a fresh copy of the same file if some users or processes request that particular file. We wanted to study this process closely, again using the `tcpdump` command. So, we launched the following command on the AFS server machine rs600030:

```
tcpdump -I -nt -i tr0 src 9.24.104.124
```

Then we logged in as afsuser on the AIX machine rs600012, and we tried to open the file we had previously modified on wtr05195. This is the output we got on rs600030 after launching the above command:

```
9.24.104.124.7001 > 9.24.104.97.7003: udp 48
9.24.104.124.7001 > 9.24.104.97.7000: udp 44
9.24.104.124.7001 > 9.24.104.97.7000: udp 52
9.24.104.124.7001 > 9.24.104.97.7003: udp 61
9.24.104.124.7001 > 9.24.104.97.7000: udp 61
```

First of all, the AFS client machine rs600012, from its port 7001, contacts the Volume Location Server process, vlserver, listening on the port 7003 on the AFS server machine. The reason for this call is that the Cache Manager knows that the file has been changed, and needs to know from the Volume Location Server the exact location of the file. In fact, the Volume Location Server maintains the Volume Location Database, which is aware of which File Server Machine in the cell houses the volume that contains the file you are searching for.

After this first call, the AFS client machine knows the exact location of the file, and physically retrieves a fresh copy of it from the File Server Machine.

Chapter 3. Caching and Filtering Component

This chapter gets very specific about the Caching and Filtering component of IBM WebSphere Performance Pack Version 1.0, also known as IBM Web Traffic Express 1.1.

- It is a proxy server, which means that it acts as a gateway and assumes the responsibility for retrieving Internet data for multiple browser clients. Client requests are sent to the Web servers through the proxy. In other words, the client is configured to send its request to the proxy first, and then it is the proxy that forwards the client's request to the Web server, acting on behalf of the originating client. The server does not even see the IP address of the client, but only the one of the proxy server. Once the proxy receives the information from the content server, it forwards the information to the requesting client. This way the responsibility and the machine load associated with making the URL request, is transferred to a dedicated machine.
- It has a caching functionality, which means that it can save, or cache, all the Web documents it retrieves and serves subsequent requests for those documents from its local cache. The client gets the information faster and network bandwidth is reduced.
- It has a content filtering functionality, implemented through the Platform for Internet Content Selection (PICS) labels, that rate Web material by criteria such as language, nudity or violence. This is a consistent way to implement filtering on a broad scale and control the content you are providing. You can apply these filters in addition to, or instead of, the filtering set by the browsers. It is especially useful where end users do not (or should not) have access to those controls.

The Caching and Filtering component of IBM WebSphere Performance Pack also has the functionality of a Web server, which is very useful for administrative purposes. In fact the administrator of the caching and filtering proxy server can access the configuration files also through the Configuration and Administration Forms, which can be accessed through the Web. However, we discourage you from using this component as a pure Web server. Although this would be possible, a more powerful Web server, such as Lotus Domino Go Webserver, is recommended, especially for its capability of being integrated with IBM WebSphere Application Server.

Notice that client browsers can be configured to automatically send their request to a caching proxy server. When the proxy server receives a request, it first attempts to serve the request from the cache. If the requested document is available in the cache, the Web server returns it to the Web client. If the document is not in the cache or is old, the proxy server forwards the request to the appropriate content server. When the proxy server receives the response, it relays the response to the Web client. If the proxy server determines the response can be cached, it stores it, so that it is available for subsequent requests.

The Caching and Filtering component of IBM WebSphere Performance Pack:

- Receives requests from Web clients
- Serves requests for documents from the cache, if possible

- Fetches documents from destination servers if they are not available in the cache
- Manages the cache of documents
- Allows content filtering at the server level

3.1 Features of the Caching and Filtering Component

The Caching and Filtering component of IBM WebSphere Performance Pack allows you to minimize network bandwidth and ensures that your end users spend less time repeatedly retrieving the same pages from the target servers. It allows you to customize its caching features to your own benefit. You can specify which pages are cached, when the information on a page will expire, how large to make the cache and when to update it.

Key features of advanced caching include:

- The ability to handle very large caches (over 10GB).
- Enhanced caching algorithm that accommodates the variable size and arrival characteristics of Web objects in order to optimize your end user response time.
- An option to automatically refresh the cache with the most frequently accesses pages. These can be identified by an administrator or determined by a system from the cache logs.
- An option to cache specific pages, even when their header information says to fetch them every time.
- Configurable cache maintenance that includes a daily, rather than continual, garbage collection for improved server performance.

Other important features of the Caching and Filtering component of IBM WebSphere Performance Pack include:

- Content filtering using the PICS labels. This is a very useful feature, since proxy server controlled filtering is more effective than browser-based filtering. PICS filtering at the proxy server level is transparent to the client and the user cannot compromise it, while PICS filtering at the browser level can be compromised very easily, for example re-installing the browser. The administrator can specify the filter rules to be enforced by the caching and filtering proxy server administrator.
- Flexible-client SOCKS that allows requests for specific IP addresses to go directly to the destination server instead of being routed through the SOCKS server.
- An option to increase client anonymity by configuring the server to strip or modify HTTP header information, `From:` fields and client IP addresses.
- A Proxy Activity Monitor that provides summary information and recent entries from the cache and proxy access logs. You can use this data to configure the caching features and to improve your server's performance.
- Files transferred by the FTP protocol are cached *only* when a complete file is received. Incomplete FTP files are not cached. Additionally, FTP files are maintained in the cache in the same way as HTTP files. The Caching and Filtering component of IBM WebSphere Performance Pack generates a

`Last-Modified`: header based on the file date of the file which is obtained from an FTP directory listing and uses this date to calculate an expiration time for the file. When a cached FTP file expires, the caching and filtering proxy server compares the current file date for the file, which is obtained by doing an FTP directory listing for the file, to the FTP directory listing file date obtained earlier, and decides whether to serve the cached file or request a new one from the FTP server.

3.2 Why Do I Need a Caching and Filtering Proxy Server?

The Caching and Filtering component of IBM WebSphere Performance Pack provides a valuable and scalable solution to some of the major traffic management problems:

- Costs and constraints on network bandwidth, particularly during periods of peak concurrent activity
- Scalable infrastructure that provides cost-effective growth paths and essentially unlimited capacity potential with minimum redesign or disruption
- Bandwidth management capabilities based upon content filtering and proxy functions
- Content management capabilities based upon industry-standard filtering technologies to restrict information or to enhance information provided to users
- Functional openness to permit evolution and exploitation of emerging Internet capabilities with minimal impact of overall network architecture
- Multiple platform support to simplify implementation planning and skill requirements

3.2.1 Caching Proxy Function

The caching proxy function provided by the Caching and Filtering component of IBM WebSphere Performance Pack is valuable to customers and/or service providers needing to optimize line costs and performance associated with accessing remote Web sites. Customers that can benefit from using the caching and proxy function of a caching and filtering proxy server include:

- Service providers needing to provide good response time to clients from Web sites accessible only via expensive or distant links that carry significant propagation delay time.

They need to be able to provide non-disruptive access to information on servers located within their networks as well as those external to their networks. Many service providers must have infrastructures capable of cost-effective expansion to handle growth rates greater than 10% per month.

- Enterprises with significant external Web access (Internet and/or intranet) wanting to optimize wide area line usage.

Large corporations and university campuses are the most likely to benefit from caching, to improve response time while optimizing external links.

3.2.2 Filtering Function

The filtering function of the Caching and Filtering component of IBM WebSphere Performance Pack is particularly valuable to customers and/or service providers wanting to be able to filter out content from Web sites on the basis of defined codes, rules or APIs. Such customers include:

- Service providers wanting to enable PICS filtering for consumers or to enforce regulations prohibiting certain types of content.
- Service providers wanting to be able to filter content to suit the needs of closed user groups, such as subscribed-based communities of interest.
Such filtering would likely reflect non-standard rules, programmed to suit the preferences of a particular interest group.
- Service providers wanting to be able to filter or intercept content to enable additional processing such as premium content (authentication or billing) or language translation.
- Business network managers wanting to be able to minimize traffic on backbone network links during peak periods by filtering traffic according to application or content priorities.
- Education institutions, particularly grade schools, wanting to control student access to non-approved content.

3.3 Cache Management

After the caching feature of your caching and filtering proxy server is turned on and the cache directory has been specified, there are several parameters where the administrator can act:

- What documents are kept in the cache
- How many documents can be cached
- How long they are considered current
- How the documents are indexed
- When the cache is refreshed

This section is particularly useful for caching and filtering proxy server administrators that need to know how manage caching on their server. It also gives an overview of how caching works.

3.3.1 Controlling Which Documents Are Kept in the Cache

The Caching and Filtering component of IBM WebSphere Performance Pack is configured through a set of configuration files, which you can edit according to a well-defined syntax. In order to control which documents are kept in the cache, you must use the httpd configuration file. The fully specified name for this file is httpd.conf on the AIX platform, and httpd.cnf on the Windows NT platform. By editing the httpd file, you can specify which documents should be cached, how long they should be cached and which documents should never be cached.

You can specify which files to cache and which files to ignore with the `CacheOnly`, `NoCaching` and `CacheLoadDomain` directives. Notice, however, that some files are never cached:

- Documents that were requested through HTTP methods other than GET, such as POST or PUT
- All the documents that were obtained after authentication or payment
- All the documents that were dynamically generated by CGI-BIN scripts or Java servlets
- Any information passed on an SSL connection, because the proxy cannot decrypt the data passing through it
- Any URL containing a question mark ? in it

Some documents, even not belonging to the above categories, have a `Pragma: no-cache` directive in their header. This directive can appear in two places:

1. It can be included with the document when it is returned by the server, instructing a caching proxy not to cache the file in the cache
2. It can be included in a request from a client, instructing the caching proxy to get a fresh copy of the document from the current server, even if a copy in the document exists in the cache.

The administrator of the Caching and Filtering component of IBM WebSphere Performance Pack can override the `Pragma: no-cache` specification that is received as part of the request from the client by using the `ProxyIgnoreNoCache` in the `httpd` configuration file. If `ProxyIgnoreNoCache` is specified, and a request is received from a Web browser that specifies `Pragma: no-cache`, the `Pragma: no-cache` in the request will be ignored and the document will be served from the proxy cache, assuming it exists there.

The `ProxyIgnoreNoCache` directive has no effect on documents that are received from the content server with a `Pragma: no-cache` header directive. If a document is received from the content server with the `Pragma: no-cache` header directive, this file will not be stored in the cache, even if `ProxyIgnoreNoCache` is specified.

3.3.2 Cache Freshness

An important issue that the administrator of the caching and filtering proxy server must face is to ensure that cached documents are consistent with the original documents located at the originating Web server. In other words, the administrator must ensure *cache freshness*.

For each document that has been cached, the Caching and Filtering component of IBM WebSphere Performance Pack computes a time at which the document will expire:

- For HTTP documents, it is the header of the document generated by the Web server that contains the expiration information.
- For FTP documents, as we have already mentioned (see 3.1, “Features of the Caching and Filtering Component” on page 108), it is the Caching and Filtering component of IBM WebSphere Performance Pack that generates its own `Last-Modified:` header information to compute expiration times. The reason for this is that the FTP protocol does not include expiration information, unlike the HTTP protocol.

The content server can indicate the expiration time in several ways, putting header information in the HTTP response. The permitted header information is described in the following list, in order of preference:

1. The content server can specify the time the document is good after it has been received. This can be done through the following header information:

```
Cache-Control: max age=xyz
```

where *xyz* is the number of seconds after which the document must be considered expired.

Pages that have been dynamically generated, for example through a CGI-BIN script or a Java servlet, may include header information saying:

```
Cache-Control: no-cache
```

2. The content server can specify the exact time at which the document should be considered expired. The header information in this case has the following syntax:

```
Expires: xyz
```

where *xyz* this time is the time at which the document expires.

Dynamically generated pages that frequently are not cacheable, may include a header saying:

```
Expires: 0
```

3. The content server can specify the time *xyz* when the document was last modified, using the following header information:

```
Last-Modified: xyz
```

Then the caching and filtering proxy server performs the following sequence of operations:

1. It computes how long it has been since the document was last modified (for example, 5 days).
2. It multiplies this number by the value (for example, 0.2) of the `CacheLastModifiedFactor` variable.
3. It assumes the document will be good for that long. (In our example, the result is 1 day.)

Notice that almost all static Web documents include `Last-Modified: xyz` header information. This is in fact the most common way that proxies use to compute expiry time for documents.

Notice that if none of the above header information has been specified by the content server, then the Caching and Filtering component of IBM WebSphere Performance Pack will look in the `httpd` configuration file for the `CacheDefaultExpiry` directive that matches the current URL and uses that for the expiration time.

CacheDefaultExpiry Directive

When setting the `CacheDefaultExpiry` directive in the `httpd` configuration file, you should be careful when setting its value to other than 0 minutes for `http:` URLs. In fact many dynamically generated pages include none of the headers mentioned above, so in those cases the value of the `CacheDefaultExpiry` directive applies. Setting the value of the `CacheDefaultExpiry` directive for an `http:` URL to more than 0 minutes will allow the proxy to cache those pages, but in this way end users may get out-of-date content.

While it is permitted to code something other than 0 minutes, you would only want to do this under specific circumstances where you explicitly wish to cache dynamically allocated pages and intentionally return out-of-date content. You would not normally set this value to other than 0 when setting up your caching and filtering proxy server.

Another important directive that is found in the `httpd` configuration file is `CacheMinHold`. After the expiry time has been calculated as described above, the Caching and Filtering component of IBM WebSphere Performance Pack checks the `httpd` configuration file to see if there is a `CacheMinHold` directive that applies for the current URL. This directive also specifies an expiration time. If such a directive is found, then the maximum time is computed between the expiration time previously computed and the value of the `CacheMinHold` directive. The result of this operation is known as the *final expiry time*.

The final expiry time is then compared to the time specified in the `CacheTimeMargin` directive. If the final expiry time is greater than the time specified by the `CacheTimeMargin` directive, then the document is cached. Otherwise it will not be added to the cache.

When a document is found in the cache, but it is expired, the Caching and Filtering component of IBM WebSphere Performance Pack issues a special request to the content server. This request is known as `if-modified-since`. The content server sends back the document to the caching and filtering proxy server only if such a document has been modified since it was last received by the proxy. Otherwise, the content server only sends a message indicating that the document has not been modified, and does not send the entire file. At this point, the caching and filtering proxy server can serve the page to the client.

3.3.2.1 Controlling Cache Freshness

In order to control cache freshness, some directives have been created in the `httpd` configuration file, as explained in the following list:

- To manage your caching and filtering proxy server's cache, you can use `CacheDefaultExpiry`, `CacheLastModifiedFactor`, `CacheMinHold` and `CacheTimeMargin`.
- The `CacheUnused` directive allows you to specify how long to keep unused cached files.

3.3.3 Cache Size and Garbage Collection

Disk space and file maintenance are common concerns when using a cache. The Caching and Filtering component of IBM WebSphere Performance Pack allows you to control the amount of disk space used for the entire cache, as we see in 3.8.4, “Enabling Basic Caching” on page 158.

Another important feature implemented in the Caching and Filtering component of IBM WebSphere Performance Pack is the nightly cleanup process known as *garbage collection*. This process examines the files in the cache directory and attempts to remove old, expired or unused files to make room for more current files. We see more details on the garbage collection configuration in 3.8.5, “Garbage Collection” on page 162.

There are two algorithms that the garbage collection process can use when deciding which files to remove and which files to keep in the cache. One algorithm maximizes the cache to improve user response time and the other maximizes the cache to minimize network bandwidth.

To set the preferred algorithm, you can use the `CacheAlgorithm` directive in the javelin configuration file (`javelin.conf` on AIX and `javelin.cnf` on Windows NT):

1. When you are tuning your cache to minimize response time, larger files are given a higher priority for deletion and, therefore, are more likely to be removed during garbage collection.
2. When you are tuning your cache to minimize network bandwidth, larger files are given a lower priority for deletion and they are less likely to be removed during garbage collection.

3.3.4 Cache Indexing

The Caching and Filtering component of IBM WebSphere Performance Pack implements a cache directory structure and lookup methods that are different from many other proxy servers. The Caching and Filtering component of IBM WebSphere Performance Pack creates an index of the files in the cache to keep in memory as each page is added. RAM memory is used instead of other media, so that the lookup operation and retrieval times are faster.

The index separates the cached files into a set of subcaches, and for each file in the cache the index stores in memory the file name, URL and expiry information. For this reason, the RAM memory required is directly proportioned to the number of files in the cache.

When the caching and filtering proxy server receives a request from a client, the proxy checks the index in memory for that particular URL:

- If the file is not in the index, the request is made to the destination server. The retrieved URL is then checked to ensure that the document is cacheable, and the document is cached if this operation is permitted. The index is then updated with the new URL, subcache and expiry information.
- If the file is in the index, the expiry information is checked to see if the URL is stale. If the URL has expired, the caching and filtering proxy server contacts the content destination server, and the URL is replaced by the newly retrieved document with expiry and subcache information updated in the index. If the URL is still consistent, the document is served.

The cache contains shadow files that mirror the index information for the proxy to use only when the server is started. The garbage collection process updates the cached document index files.

3.3.5 Automatic Cache Refreshing

Typically, the most common proxy servers cache a particular page only after a user requests it. The Caching and Filtering component of IBM WebSphere Performance Pack, in addition to this default caching, has a *cache agent* that provides automatic caching and gives more control to the administrator. The cache agent can retrieve specified URLs even before they are effectively requested and refresh the cache automatically. The cache refresh takes place when the server activity is low (by default, every night at midnight, local time) and all the retrieved pages are ready in the cache to provide faster service even the first time a user requests them.

The automatic cache refreshing has two sources it can use to refresh the cache:

1. It can load specific URLs defined by the administrator through the `LoadURL` directive in the javelin configuration file. In this way, the administrator can specify a certain set of pages that must be loaded by the cache agent when it starts.
2. It can load the most popular URLs from the previous day's activity. To obtain this, the cache agent checks the cache access log, sorts it by frequency of requests and then picks the most popular pages. It can refresh the top number of pages as specified with the `LoadTopCached` directive in the javelin configuration file.

Notice that the cache agent can use both sources of input.

Optionally, the cache agent can follow a specified level of HTML links on the pages it is loading and cache all of those linked pages. This operation is also known as *delving*. The following figure offers a graphical representation of the delving process:

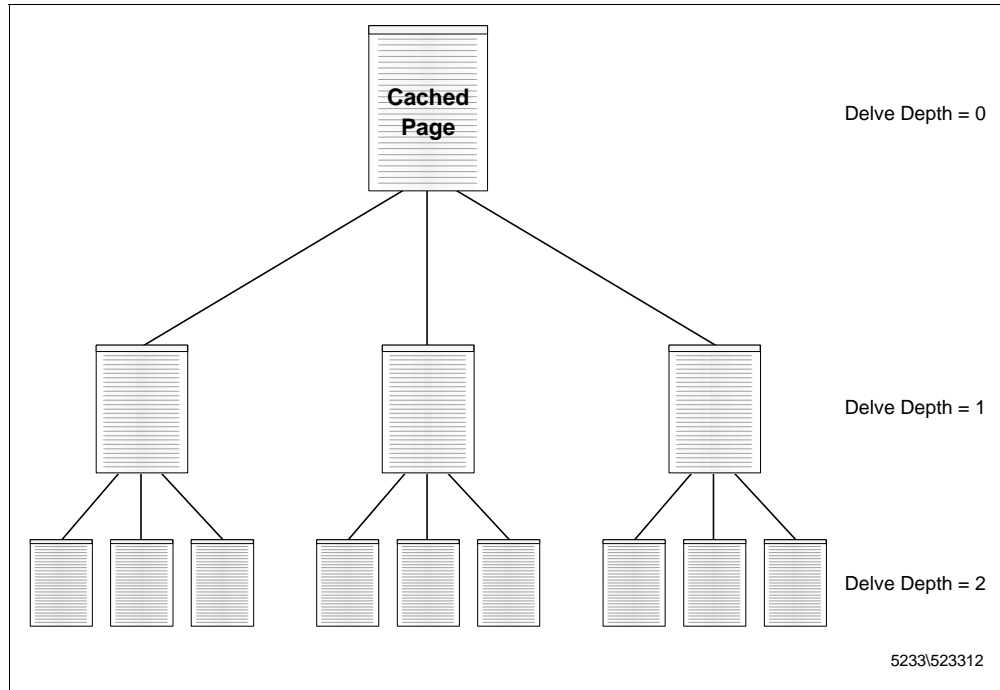


Figure 73. Delving Process

Notice that it is not necessary that the linked pages reside on the same host as the source page, since the cache agent can retrieve them even if they reside on other hosts.

The cache agent offers for sure a very useful service. Using the cache agent, caching is performed even before cached pages are effectively requested, so the average response time is minimized. Moreover the cache is built before user activity gets busy, typically at night.

However, turning the cache agent on forces the caching and filtering proxy server machine to be busy even during hours of low activity. Moreover, configuring the cache agent to perform delving requires more control to the caching and filtering proxy server administrator. For example it is recommended to disable delving from high-level pages, such as Web indexes or search sites, or multiple requests for large numbers of pages will be generated.

3.4 PICS-Based Filtering

Some browsers, such as Microsoft Internet Explorer Version 4, allow you to restrict which sites users can view. This filtering operation is performed through the PICS protocol as content is received from the destination. The disadvantage of this approach is that settings are performed at the browser itself, so that they can be easily compromised by a user that has access to the browser machine.

The Caching and Filtering component of IBM WebSphere Performance Pack has the ability to place the filtering at the proxy server level. Using a proxy to determine if a URL can be viewed removes the responsibility away from the client and places it on the provider. This method grants that the client will only get the level of content specified at the proxy.

Using this centralized approach, the PICS filtering process is transparent to all the users. When a user requests an HTML page, they will see the requested URL or get an error message similar to the following

Error 403
Blocked by Filtering Rule

This error is shown in the following figure:

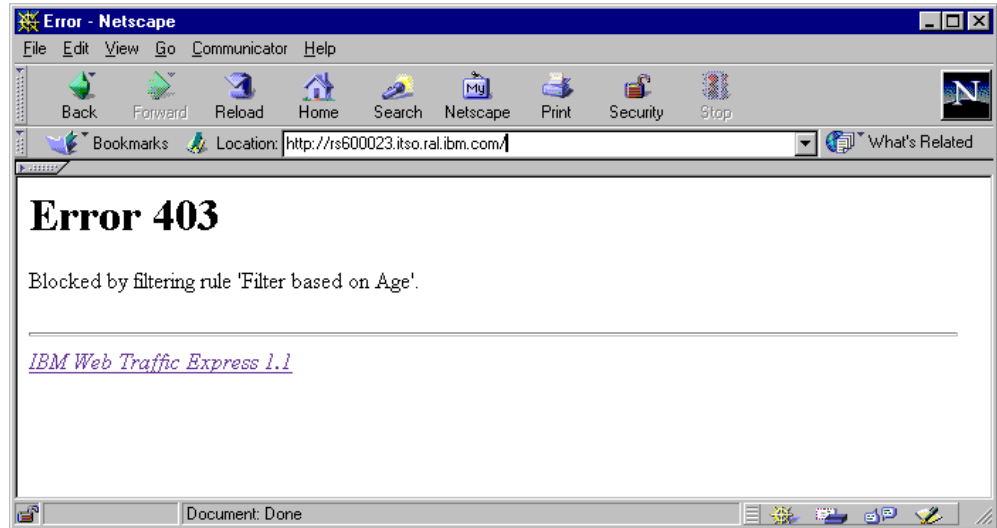


Figure 74. Error 403 with an HTML Document Blocked by Filtering Rule

The Caching and Filtering component of IBM WebSphere Performance Pack allows an administrator to specify filtering rules based on PICS labels. When a URL is accessed, the caching and filtering proxy server uses these rules to determine if it passed or failed.

The PICS labels can be supplied to the caching and filtering proxy server in several ways. They can be stored locally on the proxy's hard disk, supplied by a label bureau or even provided by the content server. Some URLs have the label embedded within their HTML files under the `<META>` tag or in the HTTP response header.

More details about how filtering works in the Caching and Filtering component of IBM WebSphere Performance Pack are shown in 3.12, "PICS Scenario" on page 186. In particular, we show you a working example where the PICS protocol is used at the proxy server level to filter Web content.

3.5 Flexible-Client SOCKS

The Caching and Filtering component of IBM WebSphere Performance Pack has a particular feature, named *flexible-client SOCKS*, that allows the caching and filtering proxy server to reside behind a firewall or SOCKS server without sharing the same physical machine. Requests going to the proxy can then be routed directly to the destination content server, instead of sending all requests through the SOCKS server.

None of the components of IBM WebSphere Performance Pack include a SOCKS server. We recommend that you install the SOCKS server provided by IBM eNetwork Firewall.

The flexible-client SOCKS functionality helps in security by allowing a firewall server to be isolated from the proxy server, even if this requires additional hardware and can produce higher latency on requests. The load on the firewall is reduced by having the caching and filtering proxy server handle internal requests. Moreover the administrator can easily specify which requests the caching and filtering proxy server sends to the SOCKS server and which requests it redirects back to the local domain.

A traditional proxy server installed behind a firewall would route all the requests to the firewall itself, as shown in the following diagram:

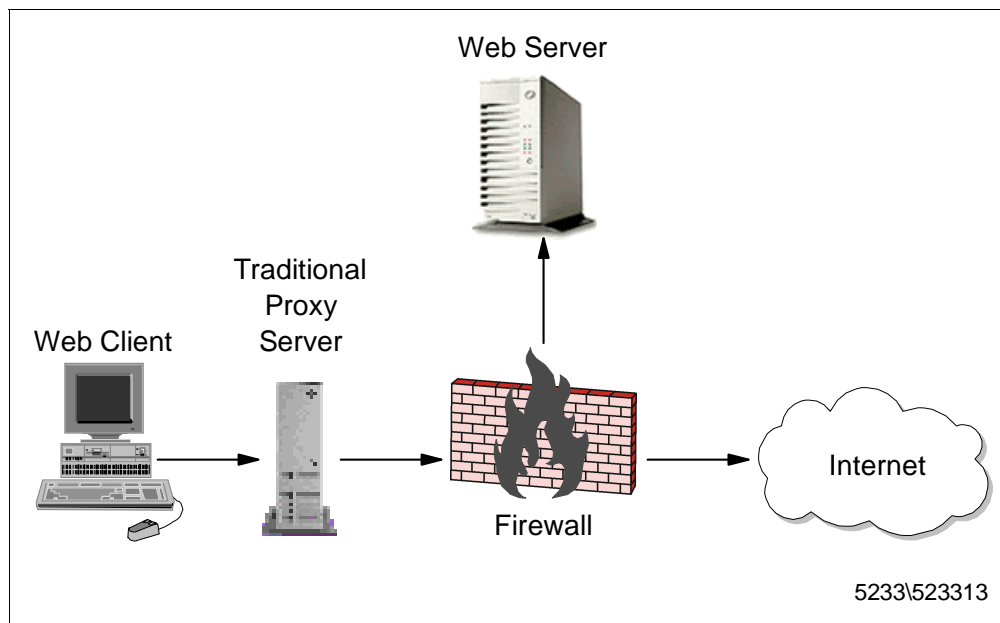


Figure 75. Traditional Proxy Server

The flexible-client SOCKS provided by the Caching and Filtering component of IBM WebSphere Performance Pack lets you specify which IP addresses or domains should be contacted directly by the proxy server and which ones should be contacted through the SOCKS server. The following diagram offers a graphical representation of this process:

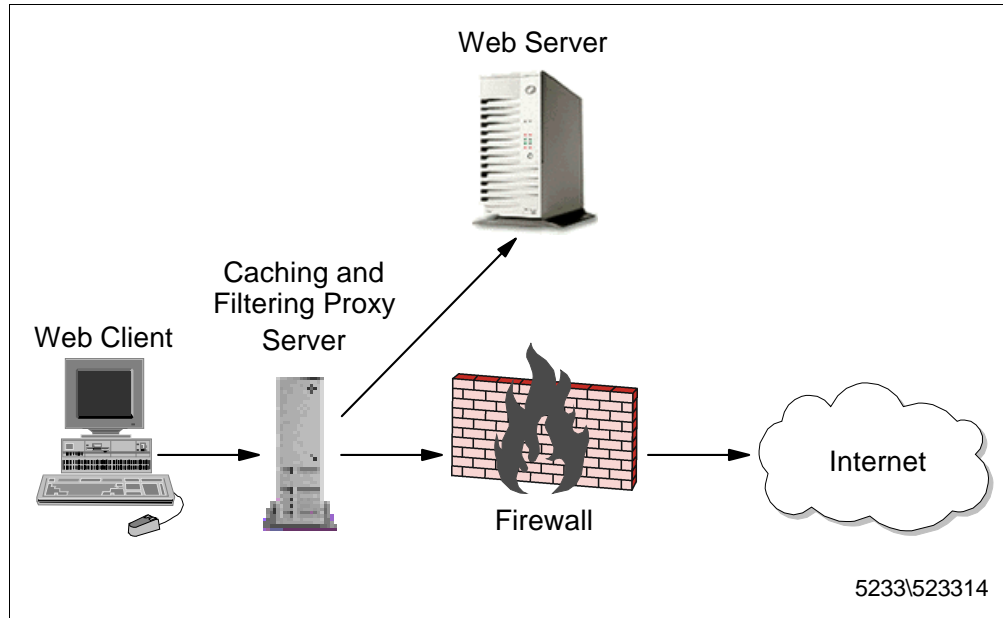


Figure 76. Flexible-Client SOCKS in the Caching and Filtering Component

If your system does not already have a SOCKS configuration file, a default SOCKS configuration file will be installed with the Caching and Filtering component. This file is named `socks.conf` on AIX and Solaris, and `socks.cnf` on Windows NT.

3.6 Other Functions

Besides the caching and filtering functionality, this component of IBM WebSphere Performance Pack offers other useful features, which we describe in the following sections.

3.6.1 Handling Header Information

The Caching and Filtering component of IBM WebSphere Performance Pack allows you to increase client anonymity by configuring the caching and filtering proxy server to strip or modify HTTP header information.

3.6.2 Proxy Activity Monitor

The Proxy Activity Monitor consists of multiple pages that contain information about the activity of the server. It provides summary information and recent entries from the cache and proxy access logs. You can use this data to configure the caching features and to improve your server's performance.

3.6.3 Caching and Filtering Proxy Server Access Protection

The Caching and Filtering component of IBM WebSphere Performance Pack can be configured to protect access to its resources, when it works as a typical Web server. It is also possible to configure it to require user ID and password authentication to all the users that try to access the proxy function, as we see in 3.10, "Proxy Server Protection" on page 174.

3.6.4 Proxy Chaining

Proxy chaining is a mechanism that allows you to create a hierarchical chain of proxies, each proxy belonging to a certain level in the hierarchy. If a proxy server in the lowest level of the hierarchy cannot serve the requested URL from its cache, it does not forward the client's request to the content Web server, but to the proxy server that has been configured as the proxy server of the immediately higher level in the hierarchy. The higher the level in the chain, the larger the number of users that access that proxy, so the possibility that a proxy server at a higher level in the chain finds the requested document in its cache becomes greater.

The proxy server at the top level of the chain contacts the Web content server to retrieve the documents if such a document was not in its cache. After that, it passes the document back down in the hierarchy, and all the proxies under it cache the document, until the lowest proxy caches the document and serves it to the requesting client.

Although proxy chaining may maximize the response time for the first requests, the average response time is minimized. All the details of how to implement proxy chaining are shown in 3.9, "Proxy Chaining Scenario" on page 164.

3.6.5 SSL Tunneling

The Caching and Filtering component of IBM WebSphere Performance Pack supports Secure Socket Layer (SSL) connections. SSL secure connections involve encryption and decryption processes and are established directly between the client browser and the destination content server. The caching and filtering proxy server does not make any attempt to cache or decrypt the information that the client and the server exchange during an SSL connection, but it establishes a connection to the destination content server and passes the requests to it without looking at the data.

This pass-through protocol is known as *SSL tunneling*. For more details on SSL and SSL tunneling, see the IBM redbook *Internet Security in the Network Computing Framework*, SG24-5220.

We explain how to enable SSL tunneling on the Caching and Filtering component of IBM WebSphere Performance Pack in 3.8.2, "Enabling Proxy Function" on page 153.

3.7 Installation of the Caching and Filtering Component

The Caching Proxy Server Component of IBM WebSphere Performance Pack is supported on three operating systems: IBM AIX 4.2.1 or later, Microsoft Windows NT Server 4.0 and Sun Solaris 2.5 or later. In this section we will show step by step how to perform the installation on AIX and Windows NT. The installation on Solaris is pretty similar to the installation on AIX. Refer to *Web Traffic Express for Multiplatforms User's Guide*, GC31-8645.

3.7.1 Installation on AIX

Before starting our discussion, it would be good to describe the hardware and software environment on which we performed our installation. The machine we used as our platform was a uniprocessor IBM RS/6000 43P having 192MB of

RAM, 133 MHz of CPU, 2.2GB of hard disk and one token-ring interface. We installed this machine with AIX Version 4.3.1.

The AIX installation program for each component of IBM WebSphere Performance Pack makes use of Java InstallShield's setup class and so it requires that the Java Virtual Machine (JVM) Version 1.1 or later is installed on the system.

We did not need to install the JVM on our machine, because the Java Development Kit (JDK) 1.1.4 fileset is automatically installed with AIX Version 4.3.1. Notice that IBM WebSphere Performance Pack requires AIX Version 4.2.1 or later.

By entering the following command we noticed that the default installation of AIX 4.3.1 locates the JVM java executable file in the directory /usr/bin:

```
which java
```

The CD of IBM WebSphere Performance Pack is provided with the JDK 1.1.4 installation file for AIX, found in the directory JDK/AIX. You need to install the JVM if it is not already installed on your system, or you might want to upgrade the level if it is previous to 1.1.4. To discover the level of the JVM already installed on your machine, you can enter the command:

```
java -version
```

The installation of JDK on AIX is described in the IBM redbook *Network Computing Framework Component Guide*, SG24-2119. Notice that the installation of the JDK from the CD does not work, because no .toc file is provided and smit cannot write one to a CD, so you have to copy that directory over to your AIX machine and do the install from there. Then you will probably want to delete the install images.

Paging Space

One of the installation prerequisites is that your system should have a minimum of 128MB of paging space. To check the amount of paging space on your system, you can use the smitty utility:

1. From a command line, enter:

```
smitty storage
```

2. Select **Logical Volume Manager** and press Enter.
3. Select **Paging Space** and press Enter to access the Paging Space screen.
4. Select **List All Paging Spaces** and press Enter.

We found that in our system the paging space size was already of 192MB. If you do not have the required paging space, come back to the Paging Space screen. At this point you have two different possibilities to increase the paging space size on your system:

- To change the size of an existing paging space, select **Change / Show Characteristics of a Paging Space**.
- To add a new paging space, select **Add Another Paging Space**.

To prepare for installation, follow the steps listed below:

1. Insert the IBM WebSphere Performance Pack CD-ROM in the CD-ROM drive.
2. From a command line, enter the following commands:

```
mkdir /CD-ROM  
mount -rv cdrfs /dev/cd0 /CD-ROM
```

3. Enter `cd /CD-ROM/AIX`.
4. To start the installation program, enter `java setup`.

After a while, you will get the Welcome window. Then, after selecting the **Next** button, you are required to agree to all items of the software license agreement. If you agree, check the box **Accept all terms of the license**, then click **Next**. You will see a window displaying the IBM WebSphere Performance Pack Version 1.0 readme file. We suggest you take a look at that file as it contains several interesting information about the product. Then click **Next** again, and a dialog box will be displayed where you can select the language, but *only* for the Load Balancing component. Notice that you have to make a selection here even if you have not specified yet that you want to install just that component:



Figure 77. Load Balancing Component Language Selection

After you select the language and click on **Next**, you are prompted to enter the IBM WebSphere Performance Pack destination location. We accepted the default destination location `/public/WebSphere` and the setup program had to create such a directory, as shown in the following window:

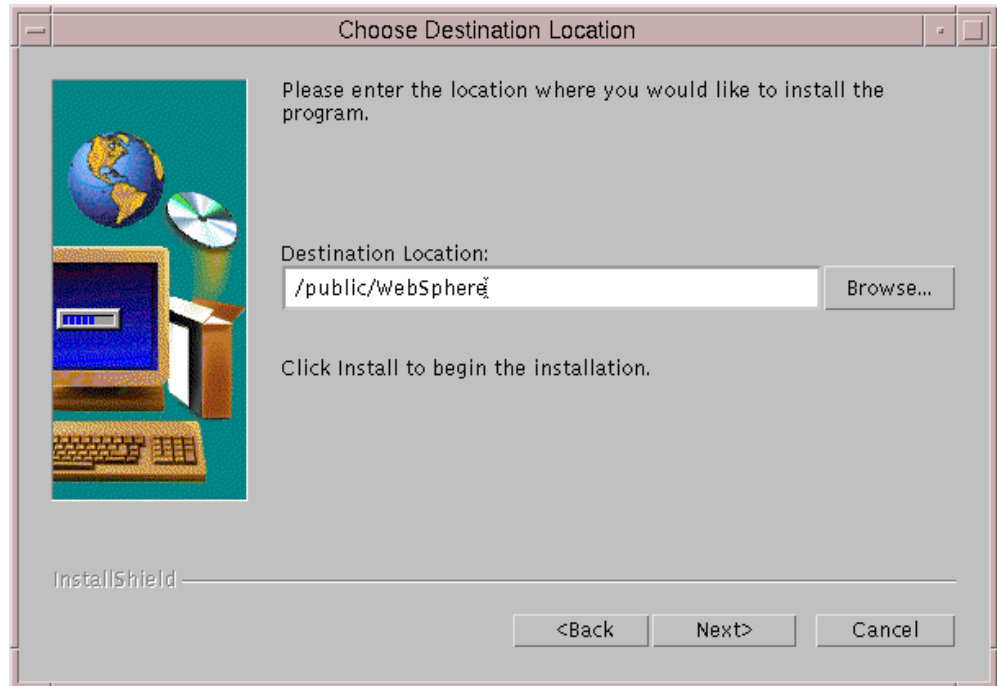


Figure 78. Choose Destination Location

Notice that in the above window you are prompted to click **Install** to proceed, but such a button does not exist. Instead, click **Next** to continue.

Select now the IBM WebSphere Performance Pack components that you want to install. Note that the installation program allows you to install on AIX these combinations of components:

- Load Balancing (eNetwork Dispatcher) component and/or Caching and Filtering component and/or File Sharing client component
- File Sharing server component and/or File Sharing client component

In this case, we selected **Caching and Filtering**, as shown in the following screen:

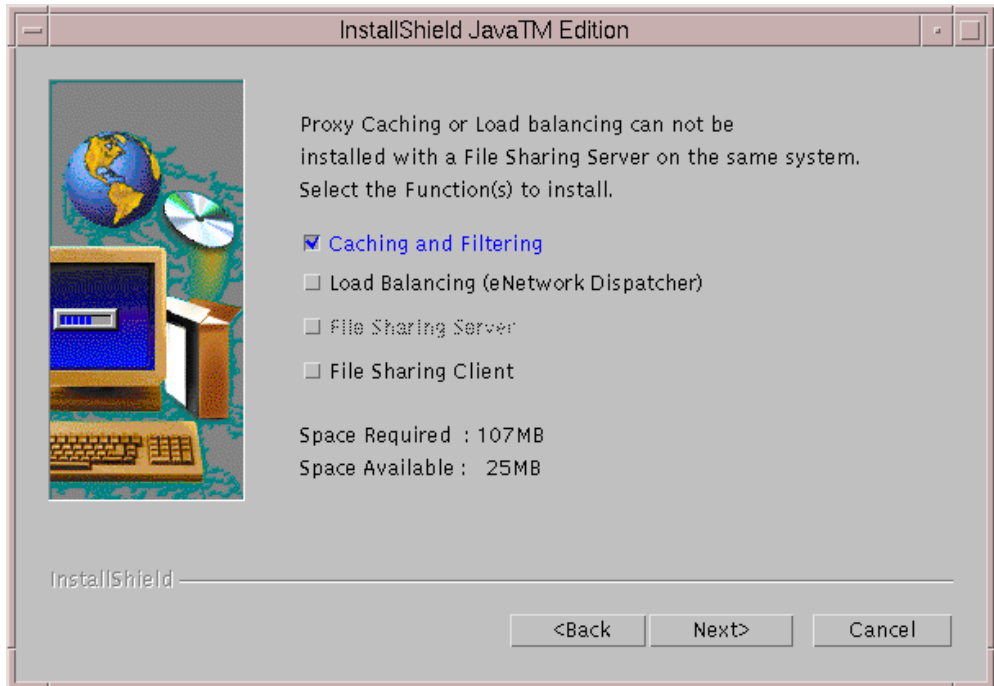


Figure 79. No Space Available for Installation

Note that we decided to install the product in the `/public/WebSphere` directory. As you can see in the above window, it turned out that the available space in the file system `/` was only 25MB: not enough to install the Caching and Filtering component, which requires 107MB. For this reason, we couldn't go on with the installation. We opened an aixterm window and we entered the command:

```
df -k
```

The output of this command displays the space available in all the mounted file systems of your systems, expressed in KB. After entering the above command, we received the confirmation that not enough free space was available in the `/` file system. Then we enlarged the file system size following the steps listed below:

1. From a command line, we entered `smitty jfs`.
2. We selected **Change / Show Characteristics of a Journaled File System**.
3. We chose the file system `/`.
4. Since about 85MB of free space are required (see Figure 79 on page 124), we added 170,000 512-byte blocks to the size of file system.

Upon doing so, we had free space enough to proceed with the installation, as shown in the following window:

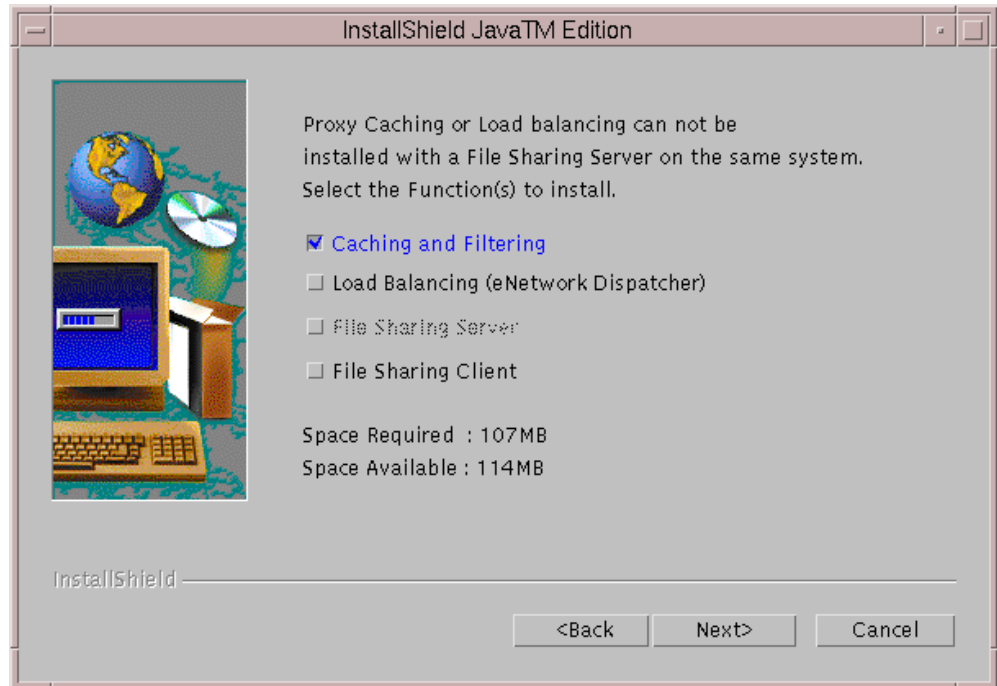


Figure 80. Free Space Enough for the Installation

After you select **Next**, you will get a window that asks you if you want the installation program to replace other programs already installed on your system. We selected **No** in this case, since we had not installed any programs on our system yet.

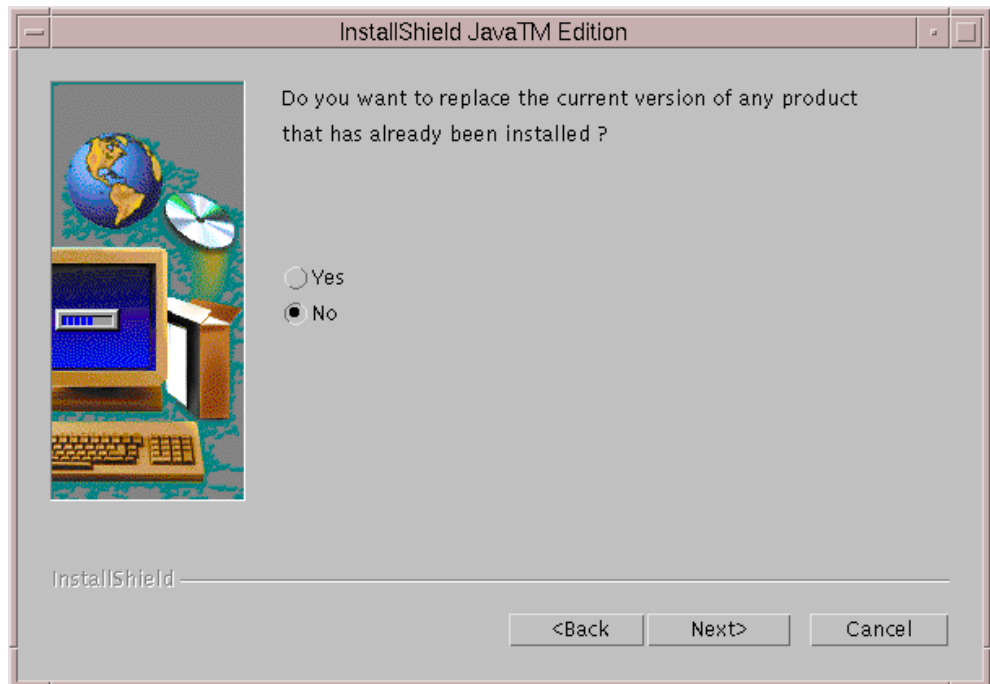


Figure 81. Choose to Replace Version

When we clicked **Next**, the following window was displayed:

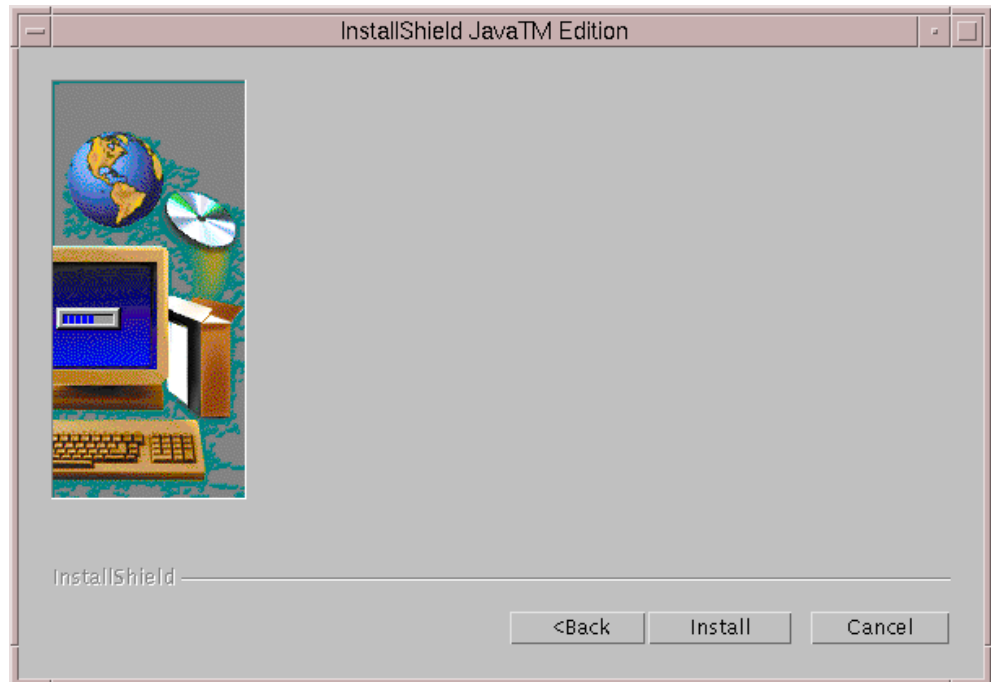


Figure 82. Start Installation Window

Click the **Install** button and after a while you will get informed that the installation is complete.

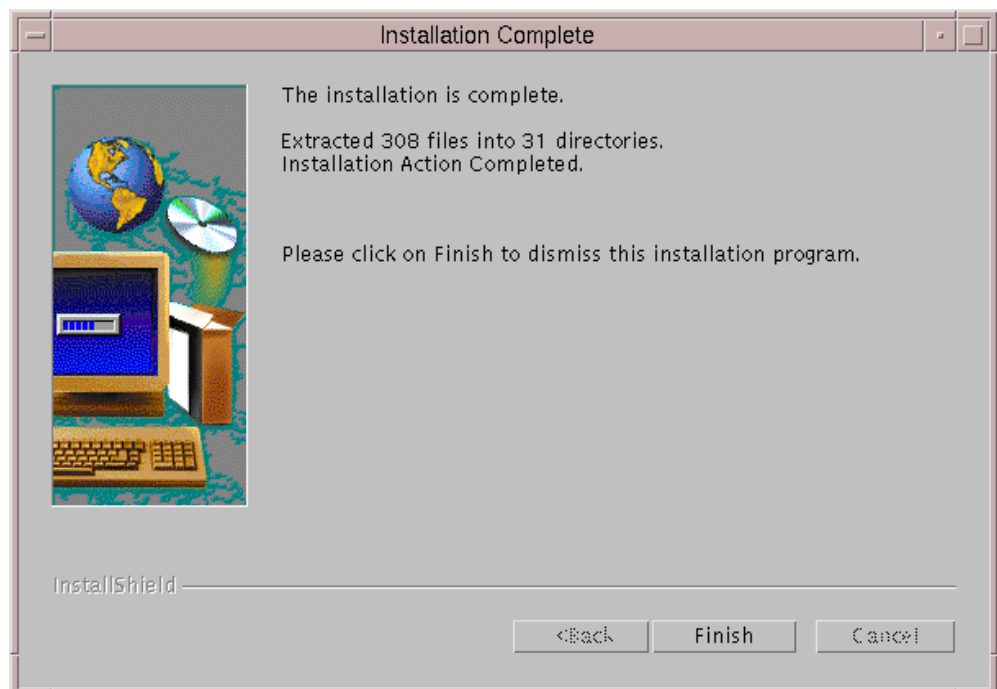


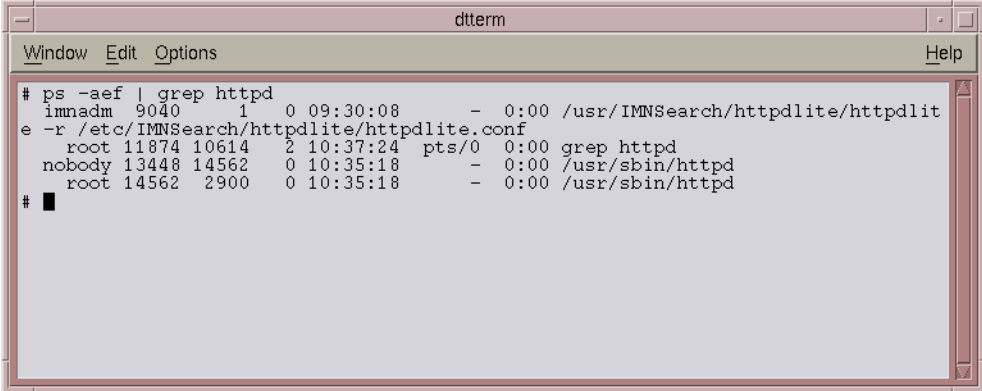
Figure 83. Installation Complete

After you finish the installation, the Caching and Filtering component of IBM WebSphere Performance Pack starts automatically with the default configuration settings. Notice that the directory where the Caching and Filtering component of IBM WebSphere Performance Pack is automatically installed on AIX is `/usr/lpp/internet/server_root`.

After selecting **Finish**, we verified that the product had been installed by entering the following command, which verifies that the `httpd` daemon is running:

```
ps -aef | grep httpd
```

If the product has been correctly installed, the `http` daemon should run at this point, and the above command should produce an output similar to the following figure:



```
dtterm
Window Edit Options Help
# ps -aef | grep httpd
imadm  9040      1    0 09:30:08    -   0:00 /usr/IMNSearch/httpd-lite/httpd-lite
e -r /etc/IMNSearch/httpd-lite/httpd-lite.conf
root  11874 10614    2 10:37:24 pts/0  0:00 grep httpd
nobody 13448 14562    0 10:35:18    -   0:00 /usr/sbin/httpd
root  14562  2900    0 10:35:18    -   0:00 /usr/sbin/httpd
# █
```

Figure 84. Checking Whether the `httpd` Daemon Is Running

Once the caching proxy server is running, you can start to configure it. To do this you can access the Configuration and Administration Forms from a Web browser. The browser can be on the same machine as the server or on any remote client that has access to the server. Remember that to use the Configuration and Administration Forms, you must previously define an administration user name and password, since for security reasons the installation process does not create any default administration user name and password.

User Names, Passwords and Groups

The proxy server maintains its own list of user names, passwords, and group names. These entities are specific to the Caching and Filtering component of IBM WebSphere Performance Pack and are not related to system users and groups of the underlying operating system.

User names and passwords are stored in the caching proxy server password file, which is `/usr/lpp/internet/server_root/protect/webadmin.passwd`. This file is empty by default after the installation and, as we mentioned, this happens for security reasons. If an entry came by default in that file, this could be a security hole, because anyone could administer their caching proxy server by simply knowing the default user name and password, unless you change the default entry as soon as you install the product. Instead the fact that the password file comes empty after the installation forces you to set an entry that only you know.

In order to add a user name and password, if `/usr/lpp/internet/server_root/protect` is the active directory, you can type the following command:

```
htadm -adduser webadmin.passwd user-name password real-name
```

The *real-name* field is optional. It should contain the name you want to use to identify the user name you are adding. The Caching and Filtering component of IBM WebSphere Performance Pack does not do anything with that field, which contains just basically a description for the administrator, but if you don't put it in, the system will prompt you for it, at which time you can just hit Enter and it will finish.

In our case, we entered:

```
htadm -adduser webadmin.passwd proxyadmin enzo1234 "WTE Server Administrator"
```

This will set proxyadmin as the user name, enzo1234 as the password and reminds you that this is the administrator of the Caching and Filtering component of IBM WebSphere Performance Pack, formerly known as Web Traffic Express (WTE). Whatever you enter will be written into the password file. This is what we found in the file `webadmin.passwd` on our system after entering the above command:

```
proxyadmin:By809y6IbIrKI:WTE Server Administrator
```

You can also make the `htadm` command prompt you for the password, if you don't type that on the `htadm` line. In this case, you are prompted twice. The password never appears in the clear while you are typing it, because the `htadm` program replaces it with a sequence of asterisks. This is good, because in this way the password does not show up in the history list. Notice also that the password is stored encrypted in the password file.

Notice that each time you install, the installation process will wipe out the `wFile Serverebadmin.passwd` file, so you have to add a new administrator user ID and password after every installation. Since before re-installing we suggest you remove the old installation, the removal would wipe out all old configuration files. For this reason, before re-installing, it is a good idea to make backups for all the configuration files and also the password file.

3.7.2 Restarting the Caching and Filtering Proxy Server on AIX

After you finish installing the Caching and Filtering component of IBM WebSphere Performance Pack, the caching and filtering proxy server starts automatically with the default configuration settings. From now on, each time you reboot your system, the `httpd` daemon will start automatically.

To stop the server, log in as root and from a command line, enter the command:

```
stopsrc -s httpd
```

You should get output similar to the following:

```
0513-044 The stop of the /usr/sbin/httpd Subsystem was completed successfully.
```

From a command prompt, you can start the server again by logging in as root and entering the command:

```
startsrc -s httpd
```

In this case the output will be similar to the following:

```
0513-059 The httpd Subsystem has been started. Subsystem PID is 12302.
```

3.7.3 Installation on Windows NT

In this section we describe all the steps that were necessary to install the Caching and Filtering component of IBM WebSphere Performance Pack on our Windows NT platform.

First of all, we describe the hardware and software environment on which we performed this installation. The machine was an IBM PC 750 with 166 MHz of CPU, 96MB of RAM, 1.5GB of hard disk and one token-ring adapter. This machine had been previously installed with Windows NT Server 4.0 and the patch Service Pack 3 had been applied.

The Windows NT installation program for each component of IBM WebSphere Performance Pack makes use of Java InstallShield's setup class. For this reason you are required to pre-install the JVM Version 1.1 or higher, which is incorporated into the JDK Version 1.1 or higher. Actually you wouldn't need the full JDK, but only its subset known as Java Runtime Environment (JRE), which contains just the JVM, the Java platform core classes, and supporting files. In other words, the JRE is the smallest set of executables and files that constitute the standard Java platform and it contains only the run-time part of the JDK: no compiler, no debugger, no tools. The CD of IBM WebSphere Performance Pack ships with the JDK 1.1.5 for Windows NT, found in the directory JDK\NT. However, we preferred to install the JDK 1.1.6 for Windows NT, since that was the latest non-beta version that was available at the time we wrote this redbook.

The latest version of the JDK can be downloaded for free from the JavaSoft Web site <http://www.javasoft.com>. The JDK 1.1.6 installation for Windows NT is very easy and so we skip its description. However, for a detailed description, you can see the IBM redbook *Internet Security in the Network Computing Framework*, SG24-5220.

The Caching and Filtering component of IBM WebSphere Performance Pack runs the setup.exe program from the CD-ROM installation directory, named NT. To do this, from the Start menu, select **Run...**, then click on **Browse...** and open the setup.exe program located in the NT directory, as shown in the following figure:

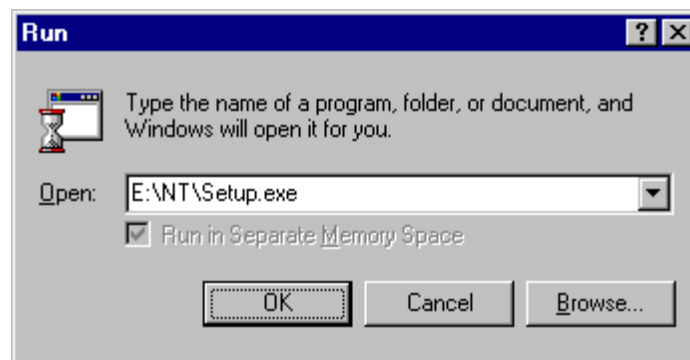


Figure 85. Installation Program's Name and Path

Notice that E in our case was the letter assigned to the CD-ROM drive.

Click **OK** to run the installation routine. It will start to find all the JVMs installed on your system. Since we had previously installed the JDK 1.1.6, the following window appeared:

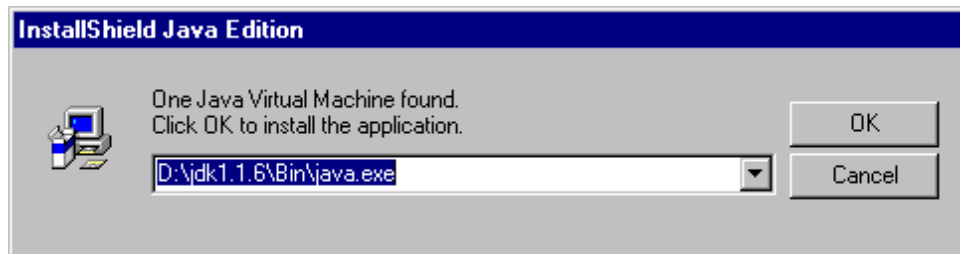


Figure 86. Selection of Java Virtual Machine Version

It's interesting to notice that if no JVMs are installed on your system, the installation routine reacts displaying the following panel:

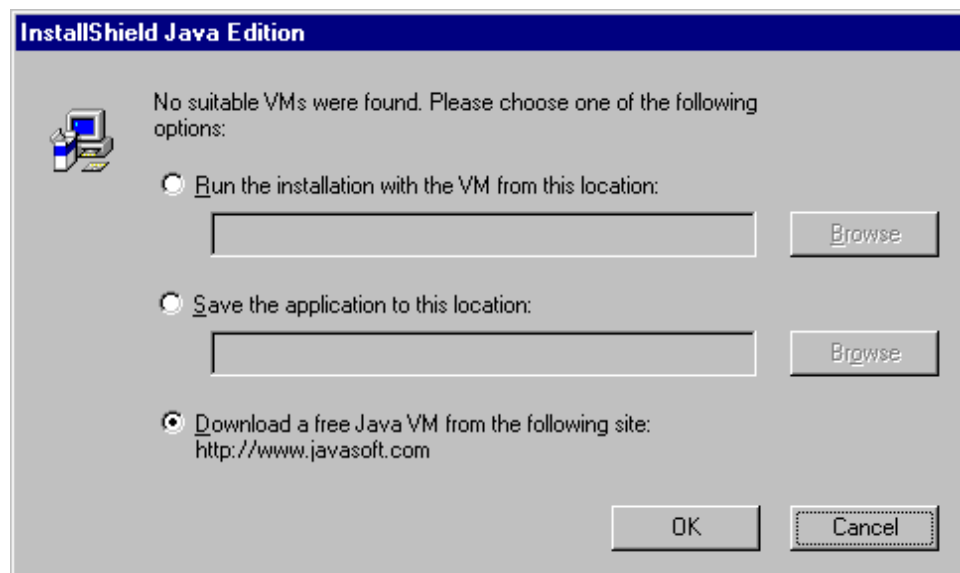


Figure 87. Choose Option If No JVM Is Installed

As you can see in the above figure, you have three options:

1. You can run the installation using a JVM located on a remote location.
If you select this option, the installation can be performed with a JVM from a remote machine.
2. You can save the Java InstallShield's setup.class on your disk.
If you choose this second option, the Java InstallShield's setup.class is copied onto a disk in your machine or in the network that your machine belongs to, but the installation is not issued.
3. You can download the JVM from the JavaSoft Web site for free.
By selecting this option, your default Web browser automatically starts and points to the JavaSoft Web site. Also in this case the installation is not issued.

Since we had already installed a valid version of JVM and the system had recognized it, we clicked **OK** in Figure 86 on page 130 and the installation started displaying the Welcome window.

After selecting the **Next** button, you are required to agree to all items of the software license. If you agree, check the box **Accept all terms of the license**, then click **Next** and you will get the following window displaying the IBM WebSphere Performance Pack Version 1.0 readme file. We suggest you take a look at that file as it contains interesting information about the product.

Click again on **Next** and a dialog will be displayed where you can select the language only for the Load Balancing component. Notice that this option is available even if you have not yet specified that you want to install the Load Balancing component, as we have seen also in 3.7.1, "Installation on AIX" on page 120.



Figure 88. Load Balancing Component Language Selection

After you click **Next**, you are prompted to enter the IBM WebSphere Performance Pack installation directory, which by default is named WebSphere. We had the setup program create it on the D drive, as shown in the next window:

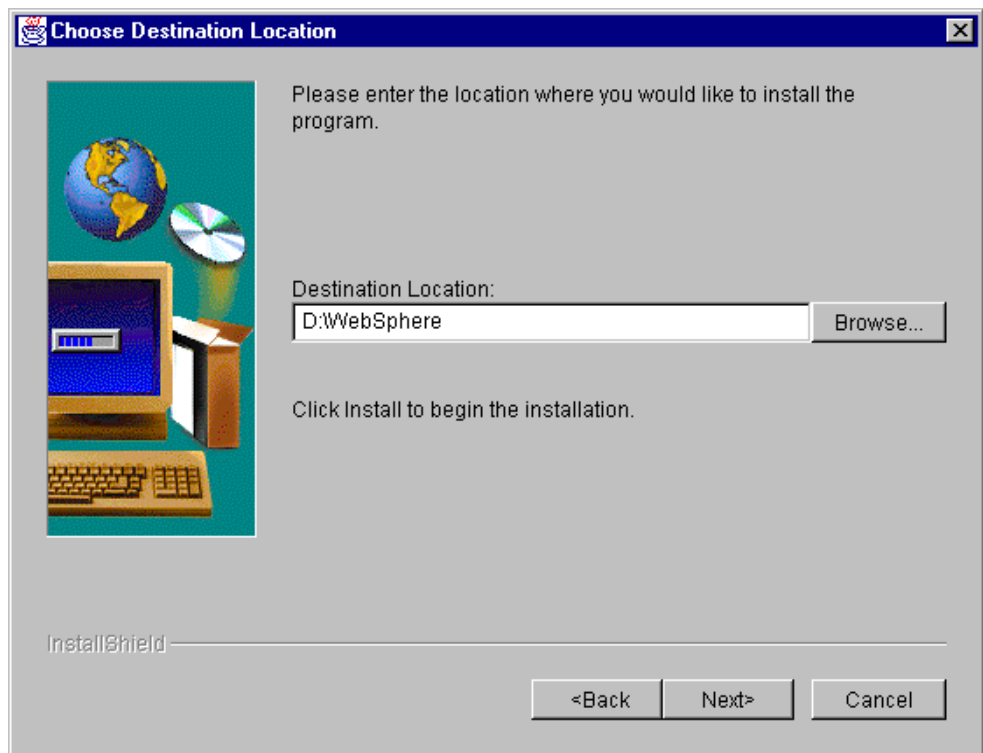


Figure 89. Choose Destination Location

Even if the above window invites you to click **Install** to continue with the installation, such a button does not exist. You should click on **Next** instead. Then you will be prompted to confirm the destination location, as shown in the following screen:

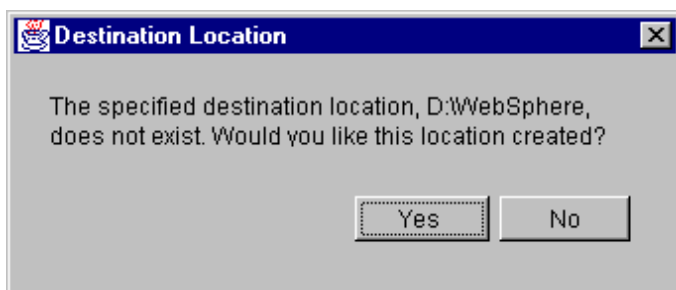


Figure 90. Forcing the System to Create the Destination Location

At this point, select the IBM WebSphere Performance Pack component that you wish to install on your Windows NT platform, as shown in the following window:

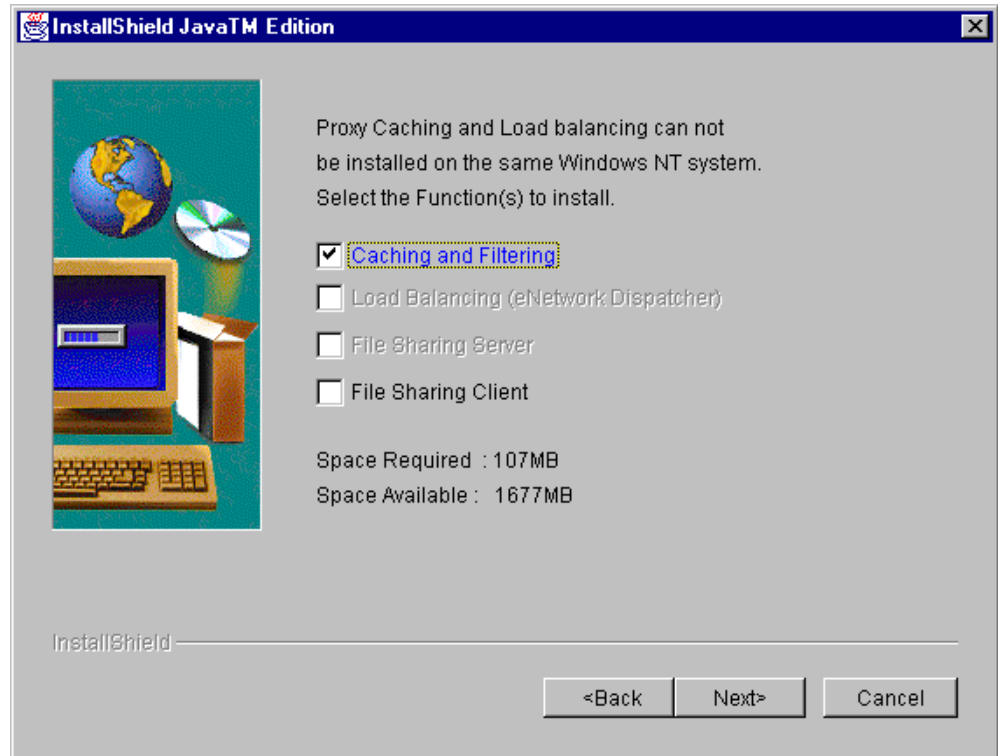


Figure 91. Selecting the Caching and Filtering Component

Note that the installation program allows you to install on Windows NT only the following component combinations:

- Caching and Filtering component and/or File Sharing client component
- Load Balancing (eNetwork Dispatcher) component and/or File Sharing client component

In other words, you cannot install the Caching and Filtering component and the Load Balancing component on the same Windows NT machine. We selected **Caching and Filtering**, and the **Load Balancing (eNetwork Dispatcher)** check box appeared immediately disabled. Notice that the File Sharing client component can be installed on the same machine together with the Caching and Filtering or Load Balancing (eNetwork Dispatcher) component.

Furthermore, note that you cannot select the check box **File Sharing Server**, since such a component is not available on Windows NT. This is the reason why the related check box appears disabled in the above window.

After you click **Next**, you will get another screen that asks you if you want the installation program to replace other programs already installed on your system:



Figure 92. Choose to Replace Version

Since we had not installed any other IBM WebSphere Performance Pack component on that particular Windows NT machine, we chose **No**. Then we clicked **Install** and the installation routine started extracting the installation files onto the disk. At the end of that process, a new window was brought up, informing us that the installation was complete, but it had not been successful, since the Runtime Installation Execution had failed:

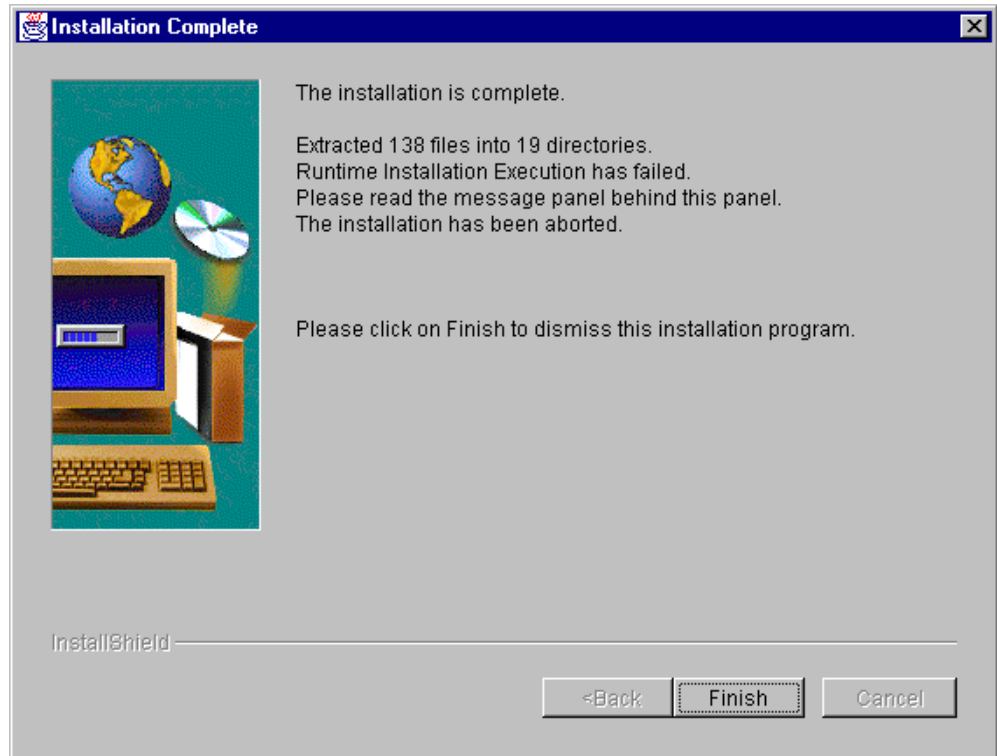


Figure 93. Installation Problem Window

You should click on **Finish** to dismiss the installation program. Notice that when a similar problem occurs, another panel is brought up on your desktop, behind the window shown in Figure 93 on page 135:

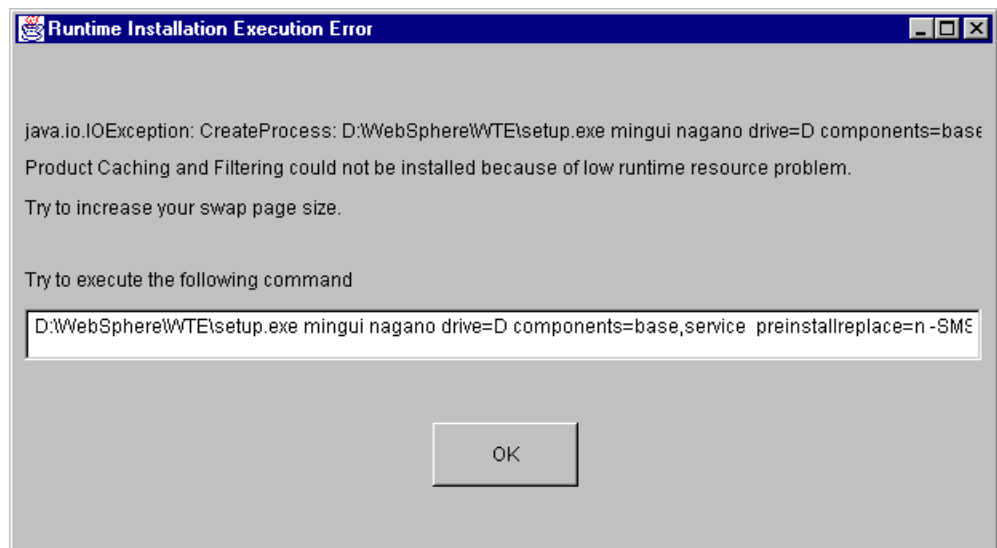


Figure 94. Runtime Installation Execution Error

This new panel suggests two ways to continue the installation:

1. Increase the swap page size of your system.
2. Execute the following command:

```
D:\WebSphere\WTE\setup.exe mingui nagano drive=D components=base,service  
preinstallreplace=n -SMS
```

On our Windows NT machine, the total paging size for all disk volumes was a minimum 200 and maximum 400MB. (In detail, we had set two pagefile.sys files in our system, one in the C drive and one in the D drive, each having a size variable between 100 and 200MB.) We also tried another installation on another Windows NT machine where the total paging size for all disk volumes was a minimum 250 and maximum 400MB. (In detail, the pagefile.sys file on that platform was 150-200MB sized in the C drive and 100-200MB sized in the D drive.) In that case the same problem occurred.

So what we did, was copy the command recommended in Figure 94 on page 135 and paste it in a Command Prompt window, as shown in the following figure:

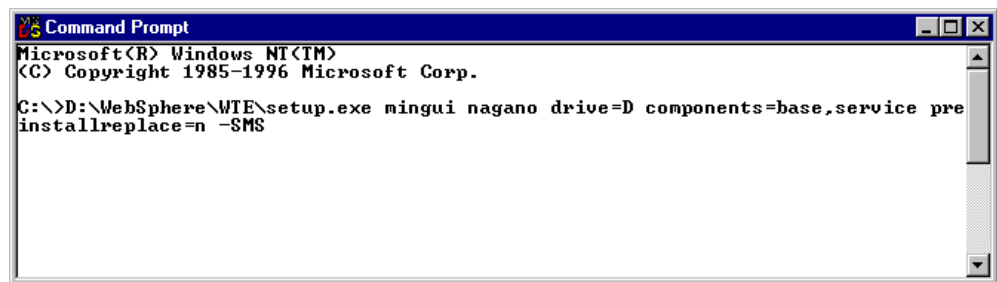


Figure 95. Installation Command

As you can notice, the above command would launch the executable file setup.exe, which the installation process copied in the D:\WebSphere\WTE directory of our system. After entering the above command, the following window is displayed, by which you can select the directory in which you want to install the files of the Caching and Filtering component:

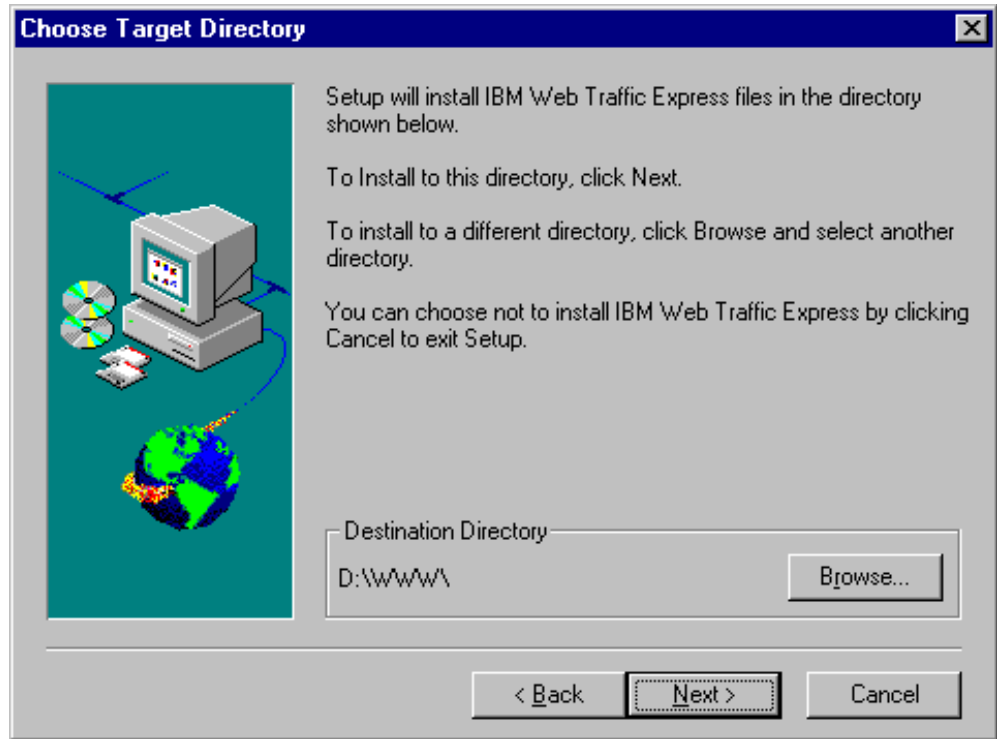


Figure 96. Choose Target Directory

The default directory is C:\WWW, but we decided to have setup.exe create such a directory under the D drive, as shown in the above screen.

We chose not to install the Java Servlet support, by clicking **No** in the next panel:

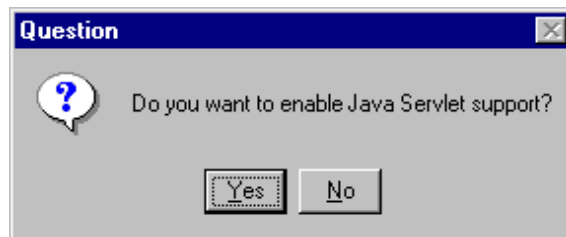


Figure 97. Enabling or Disabling the Java Servlet Support

The reason why we selected not to install the Java Servlet support was that we did not want to use the Caching and Filtering component of IBM WebSphere Performance Pack as a Web server. In fact, besides its primary function of caching and filtering proxy server, this component is capable of offering a basic Web server functionality. However, this functionality did not match the architecture we had designed for our scenarios, where separate machines running a full Web server would be involved (see Part 2, “IBM WebSphere Performance Pack Scenarios” on page 347).

After the above step, the setup program started to copy the files. At the end of the process, it may happen that the Setup window remains on the screen. The percentage bar indicates that 100% of the setup process has been completed, but the window does not disappear, as shown in the following screen:

IBM Web Traffic Express Setup

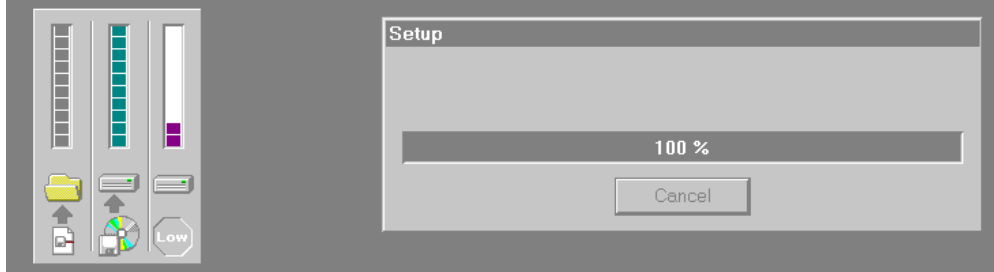


Figure 98. Problems with the Installation

If you happen to face a similar problem during the installation, we explain now how we solved it.

First of all, you should open the Windows NT Task Manager. This can be done by simultaneously pressing the keys Ctrl, Alt and Delete and then selecting **Task Manager...** in the Windows NT Security panel. In the Task Manager, we saw that the installation Setup process was in a Not Responding status. So we highlighted it and we selected the **End Task** button, as shown next:

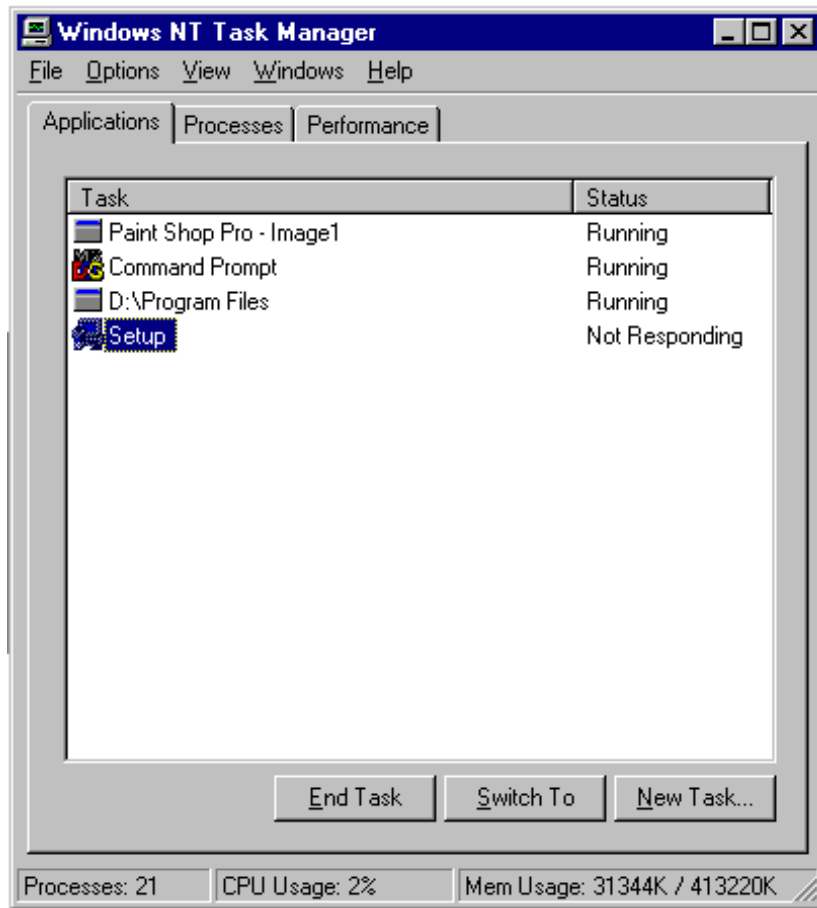


Figure 99. Stopping the Setup Process

Then we restarted the machine, as usual after a complex installation. After that, we checked the services available on our computer. To do this, we opened the Services dialog box of the Control Panel folder. We noticed that the IBM Web Traffic Express service was now in the list, even though it was not started yet, as shown in the following window:

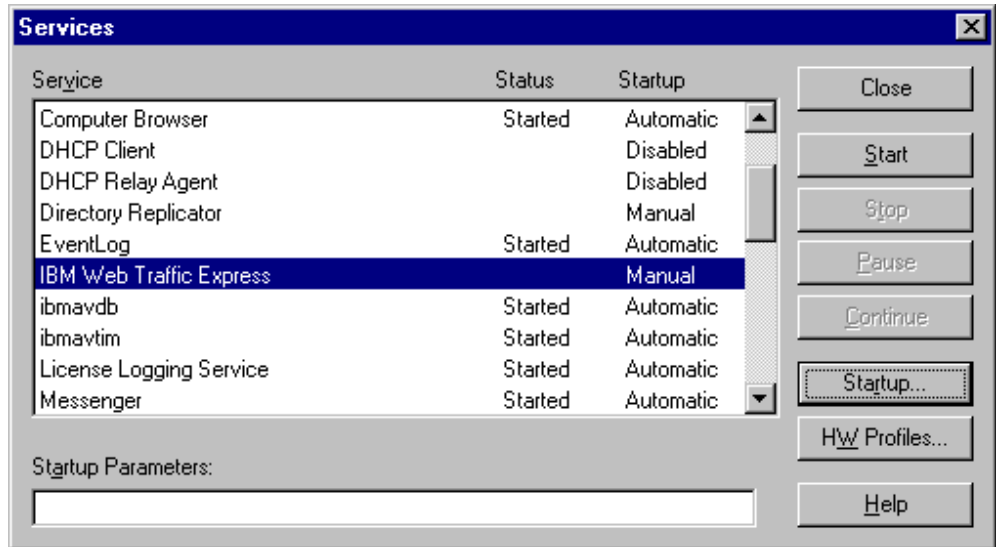


Figure 100. IBM WebTraffic Express among the Available Services

We tried to start that service by highlighting it and selecting the **Start** button, but the service did not start and we received instead an error message.

We read in *Web Traffic Express for Multiplatforms User's Guide*, GC31-8645, that in order to install the Caching and Filtering component of IBM WebSphere Performance Pack you must run the setup.exe program in order to start the installation of Web Traffic Express. For this reason, we run the setup.exe executable file, which the previous installation steps had created in the directory D:\WebSphere\Wte.

On doing so, a new installation process is activated and you should see the Welcome window. Click **Next** and you will see the Select Components window, as shown:

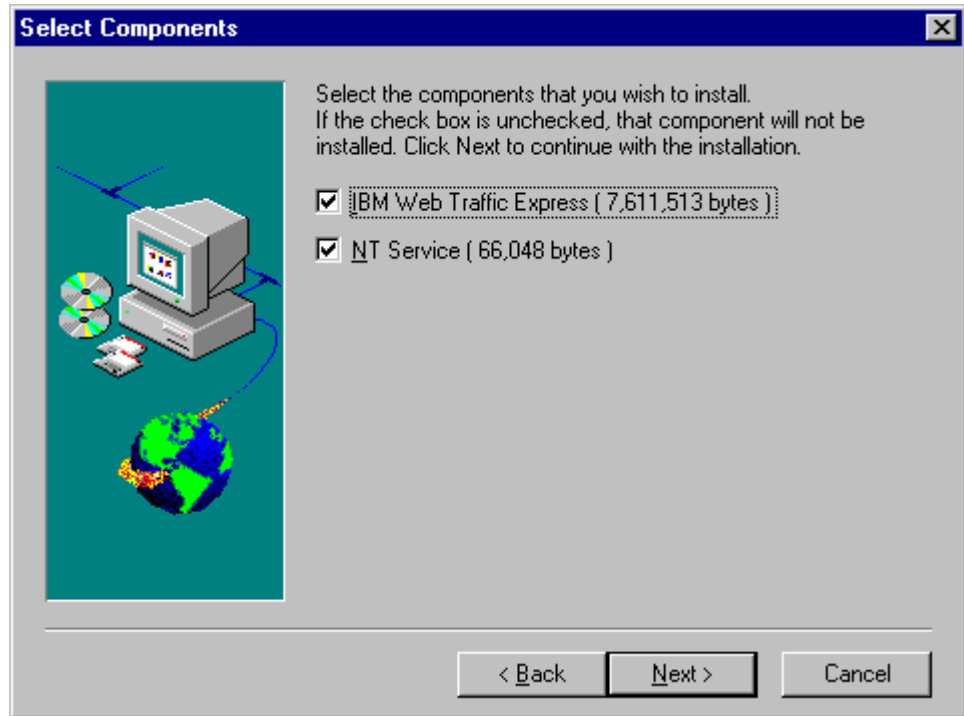


Figure 101. Select Component Window

We selected, of course, **IBM Web Traffic Express**, but also **NT Service**, which is a set of files that allow the Caching and Filtering component of IBM WebSphere Performance Pack to run as a Windows NT service rather than a simple application. Once this component is installed as a Windows NT service, the IBM Web Traffic Express program group will contain only the Uninstall IBM Web Traffic Express and ReadMe icons, because you can start a true Windows NT service only from the Services panel.

Notice that the product documentation is automatically installed.

After making your selections, click **Next** and you will get the following window to choose the target directory. Once again, the default installation directory was C:\WWW, and we modified it by clicking the **Browse...** button and selecting D:\WWW, as shown next:

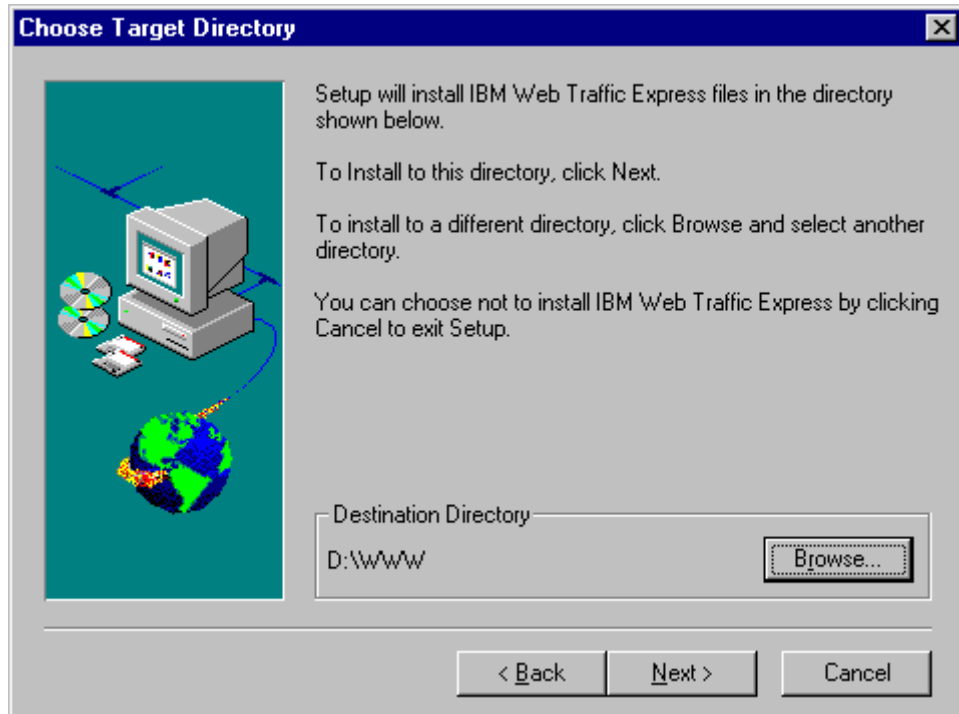


Figure 102. Choose Directory for Installation

Then we clicked on **Next** and the following window was brought up:

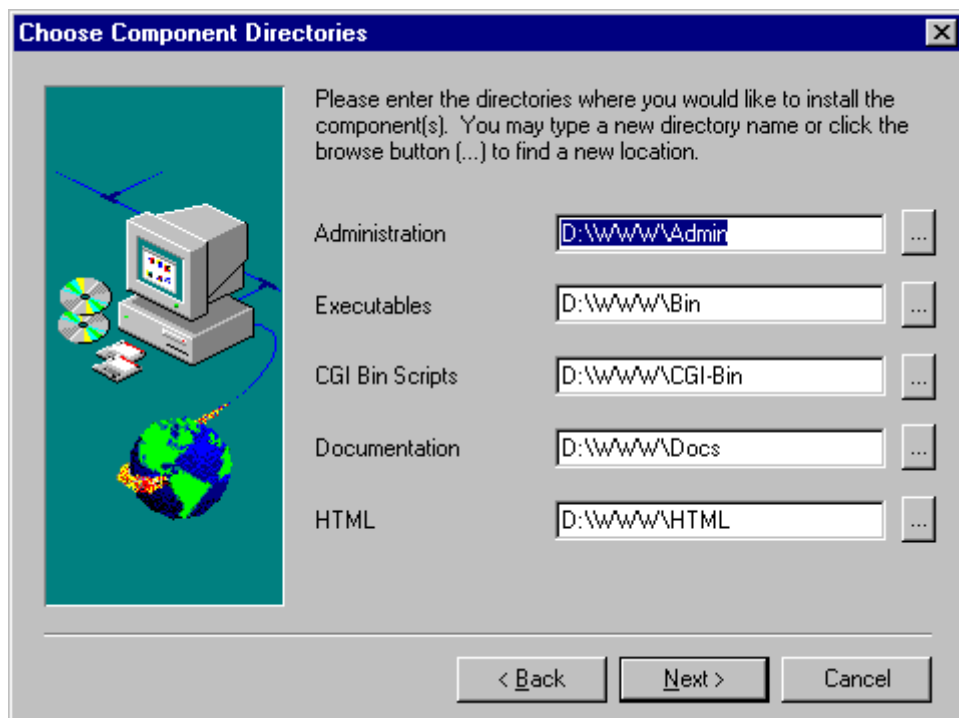


Figure 103. Deciding Where to Install Each Item

Now you can decide where to install the proxy server components. The previous dialog gives you the ability to select where you will install each of the listed individual components:

- Administration
- Executables
- CGI-BIN Scripts
- Documentation
- HTML

We accepted the default locations, then we clicked **Next** and the remaining components were displayed in a separate window:

- Icon and Graphics
- Labels
- Logs
- Servlets

You can decide where to install these items as well through the following window:

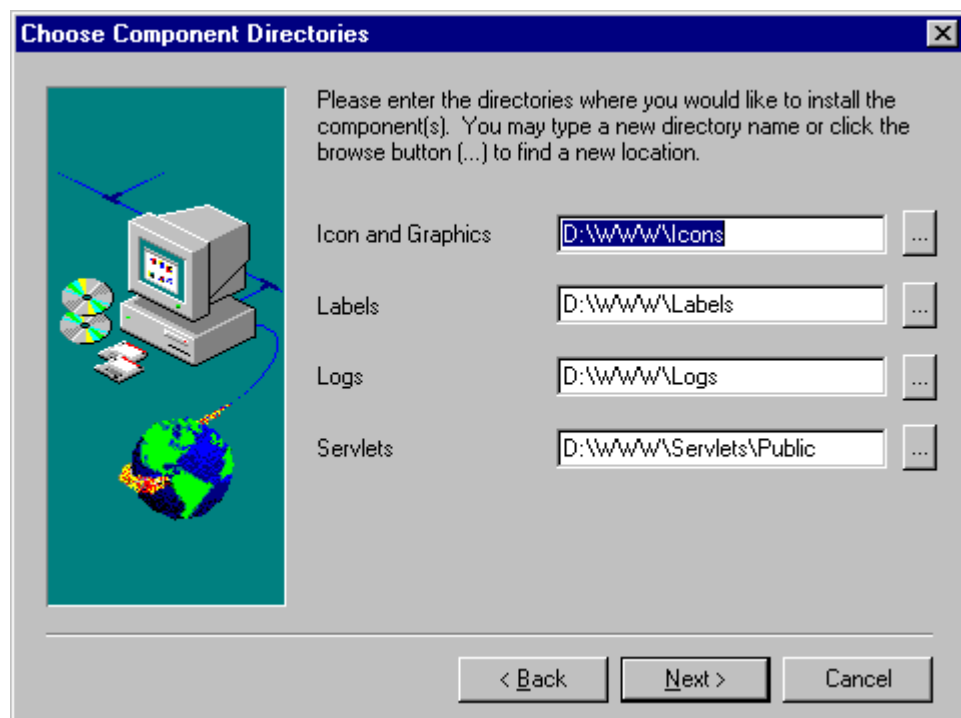


Figure 104. Deciding Where to Install Each Item

We accepted the default location for these components too, and then we clicked the **Next** button.

At this point, you are asked if you want to keep the set of existing configuration files or overwrite them with new versions. In a typical installation, such files should not be already present on your system. The reason why we found them is that we had already completed an installation process on our system. That installation process, although unsuccessful, had already placed the configuration

files httpd.cnf, ics_pics.cnf, javelin.cnf, socks.cnf and admin.pwd on the C:\WINNT directory of our system.

In order to have a fresh installation of the Caching and Filtering component, we answered **No** to the following five questions:

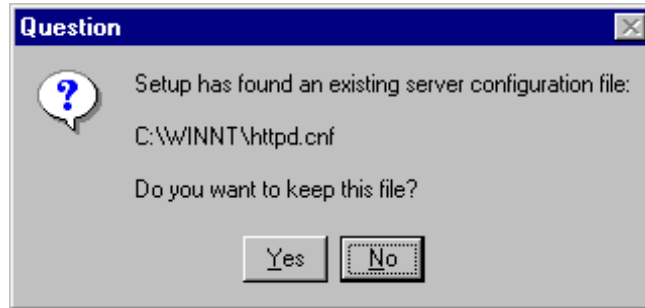


Figure 105. Reinstalling the httpd.cnf Configuration File



Figure 106. Reinstalling the ics_pics.cnf Configuration File

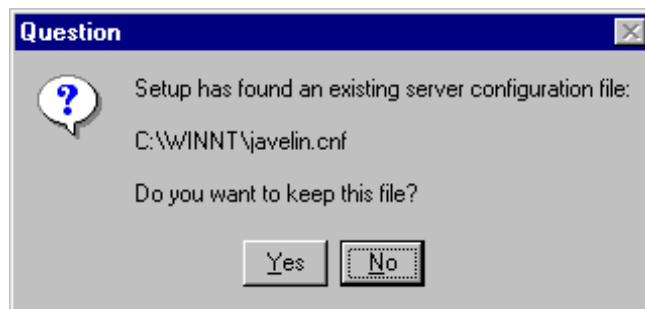


Figure 107. Reinstalling the javelin.cnf Configuration File

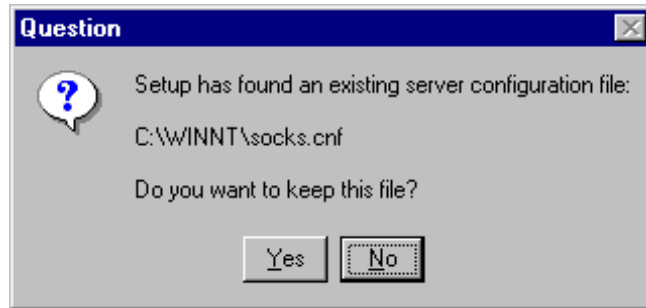


Figure 108. Reinstalling the socks.cnf Configuration File

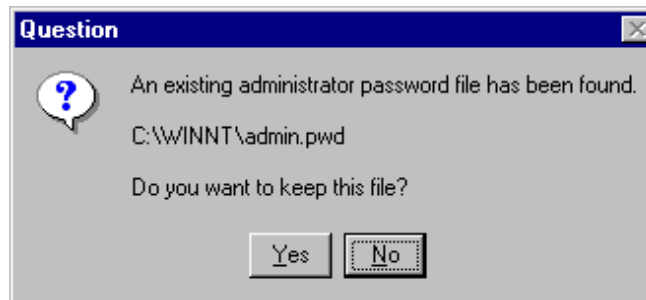


Figure 109. Reinstalling the admin.pwd Configuration File

After answering the last question, you get the following window, where you are required to enter configuration parameters for the Web Traffic Express:

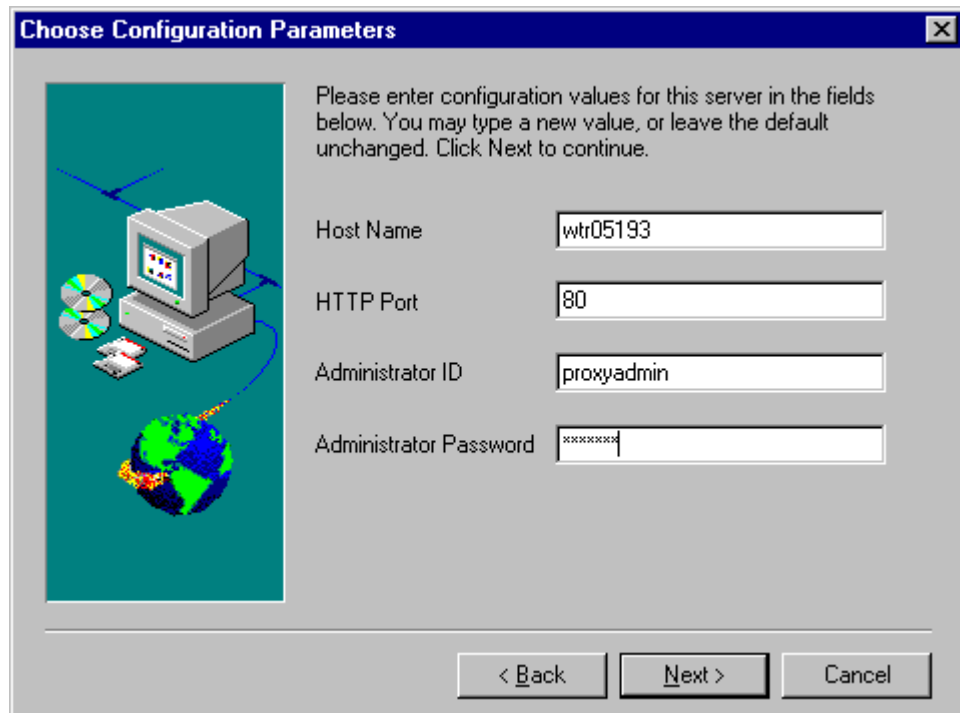


Figure 110. Configuration Values for Web traffic Express

The configuration parameters are:

- Host Name

This is the host name of the machine as it is defined in the TCP/IP configuration of your network.

- HTTP Port

This is the TCP port used on the caching and filtering proxy server machine for the HTTP protocol. We did not modify the default value 80, which is the well-known port number for HTTP.

- Administrator ID

This is the name of the caching and filtering proxy server administrator. Anyone attempting to use the proxy server Configuration and Administration Forms will be prompted to enter this ID. We entered proxyadmin.

- Administrator Password

This is the password of the caching and filtering proxy server administrator. Notice that the password is shown in clear text in the above field while you enter it, although we have shown it masked by a sequence of asterisks.

Administrator ID and Password

Notice that on Windows NT you are prompted to enter the user name and password for the proxy server administrator during the installation process, unlike on the AIX platform, where you are forced to create a user name and password for the administrator after the installation, by using the `htadm` command (see 3.7.1, "Installation on AIX" on page 120).

We want to mention here too that the caching and filtering proxy server administrator profile is stored in the directory of the Caching and Filtering component, and that it is independent on the underlying operating system. The password is stored encrypted.

Enter the parameters and when you finish click **Next**. You will be asked if you want to enable the Java Servlet support, as during the previous unsuccessful installation:

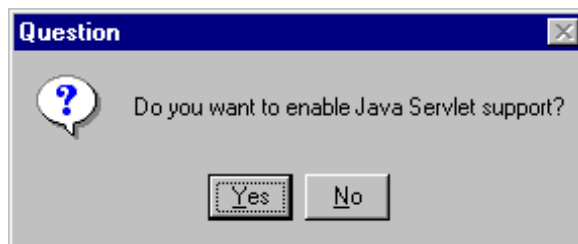


Figure 111. Enabling or Disabling the Java Servlet Support

As we have already explained, our plan was to use the Caching and Filtering component of IBM WebSphere Performance Pack as a pure caching and filtering proxy server, so we selected **No** in the above screen.

After a while, you will receive the final panel that informs you that the installation is complete and you have to reboot the system.

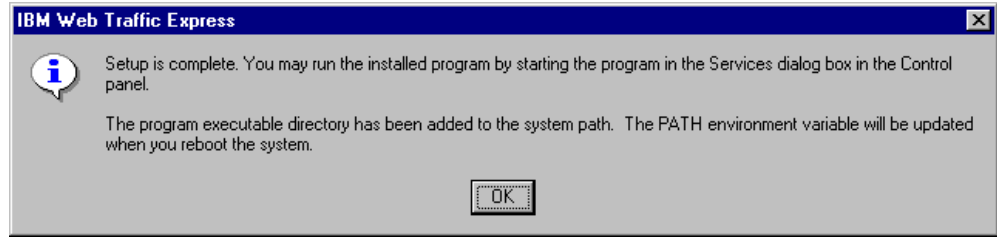


Figure 112. Setup of the Caching and Filtering Component is Complete

3.7.4 Restarting the Caching and Filtering Proxy Server on Windows NT

After rebooting the system, the proxy server is not running yet, because the service IBM Web Traffic Express has not started. If you open the Services window, you will see that the Startup mode for IBM Web Traffic Express is still set to Manual (see Figure 100 on page 140).

To start the caching and filtering proxy server, select the **IBM Web Traffic Express** service and click **Start**. The Status of the IBM Web Traffic Express service will change to read *Started*, as shown in the following screen:

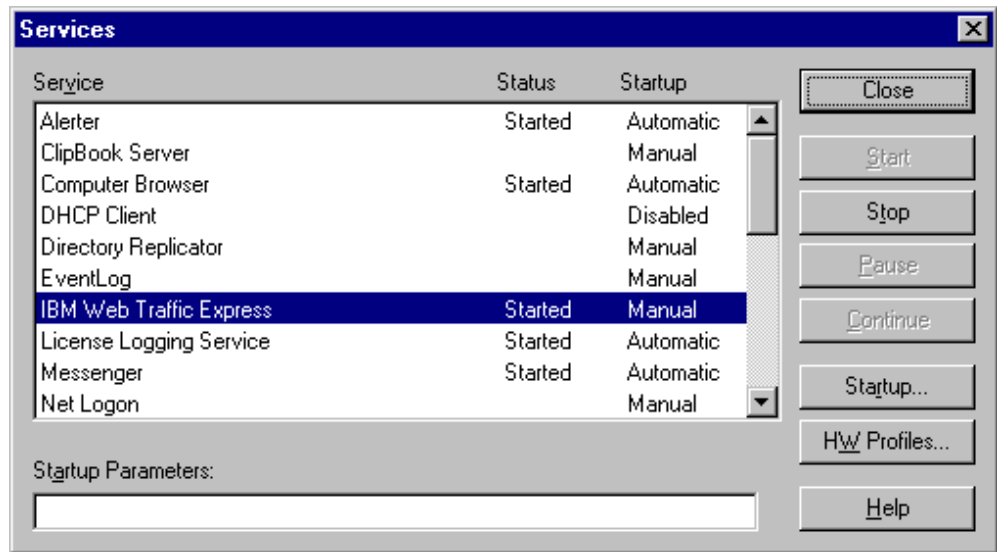


Figure 113. Starting the IBM Web Traffic Express Service

To have the above service start automatically at each reboot, make sure that such a service is selected and click the **Startup...** button. You will get the Service window for that IBM Web Traffic Express, as shown in the following figure:



Figure 114. IBM Web Traffic Express Service Window

In the Startup Type section, check off the **Automatic** radio button and click **OK**.

Normally, Windows NT services run without a graphical user interface (GUI). If you would like to see the GUI, select the **Allow Service to Interact with Desktop** check box in the above window before clicking **OK**. The GUI will appear the next time you start the service, as shown next:

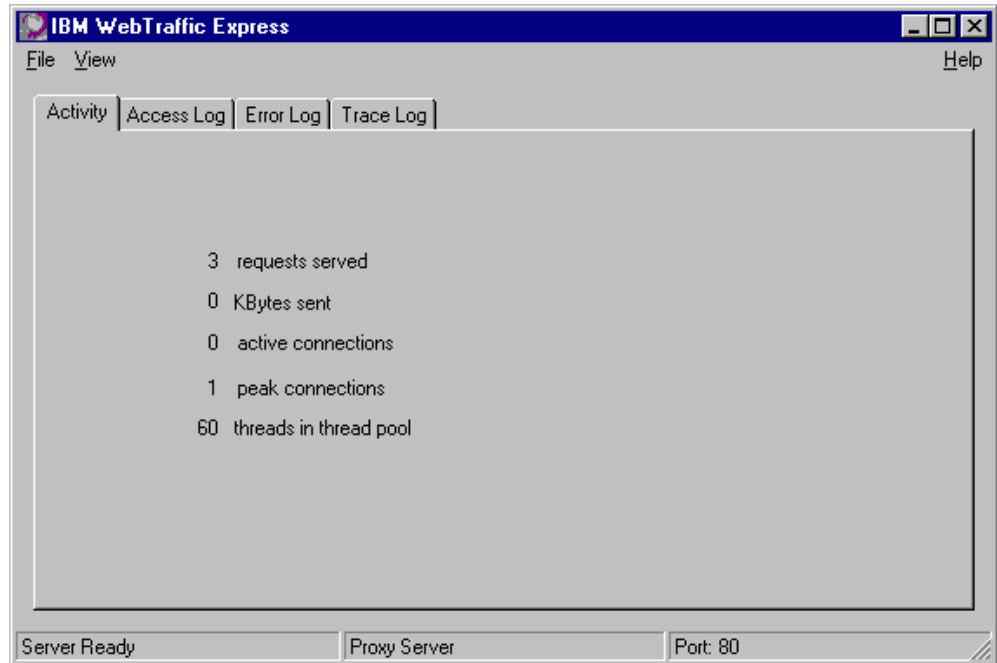


Figure 115. IBM Web Traffic Express Graphical User Interface

To stop the caching and filtering proxy server, open the Services panel again, select the **IBM Web Traffic Express** service and click **Stop** (see Figure 113 on page 147).

3.8 Basic Configuration

After you finish installing the Caching and Filtering component of IBM WebSphere Performance Pack, the caching and filtering proxy server starts automatically with the default configuration settings. This section describes the few steps you need to do to personalize the basic configuration.

The caching and filtering proxy server behavior is regulated according to some directives contained in its configuration files. We list here the configuration files and their functionality on the two platforms, AIX and Windows NT. We experimented with:

1. AIX Platform

- The proxy configuration directives are written in two files `httpd.conf` and `javelin.conf`.
- The SOCKS configuration file is `socks.conf`.
- The PICS rules are in the `ics_pics.conf` configuration file.
- If you installed Java Servlet support, you also need to edit the `servlet.conf` configuration file.

All these files are in the location `/etc`.

2. Windows NT Platform

- The proxy configuration directives are written in two files `httpd.cnf` and `javelin.cnf`.

- The SOCKS configuration file is socks.cnf.
- The PICS rules are in the ics_pics.cnf configuration file.
- If you installed Java Servlet support, you also need to edit the servlet.cnf file.

The five mentioned configuration files will by default be located in the directory defined by the ETC Windows NT system environment variable if this variable exists, or in the Windows NT system directory (typically C:\WINNT) if the ETC system environment variable is not defined.

Why Javelin?

Javelin is the old internal name of the Caching and Filtering component of IBM WebSphere Performance Pack.

As you can see, the configuration files have the same names on both the platforms, but the extensions are different: conf is their extension on AIX, while on Windows NT the configuration files carry the cnf extension. From now on, we refer to those files using just their names (for example, httpd or javelin); if we want to refer to a specific platform, we use the fully qualified names, which include the appropriate extension.

There are two ways you can set or modify the configuration of the caching and filtering proxy server:

- Editing the configuration files and modifying the configuration directives
- Using the Configuration and Administration Forms from a Web browser after authenticating as the caching and filtering proxy server administrator

We adopted the second method, because it provides a combination of HTML forms and CGI-BIN programs that build a user interface very simple to interact with. So now we show you the steps we followed through the Configuration and Administration Forms, but we also describe what the corresponding directives would be in the configuration files.

We performed the configuration of the Caching and Filtering component of IBM WebSphere Performance Pack on the same AIX machine we used to show the installation process (see 3.7.1, "Installation on AIX" on page 120). The configuration of this component on Windows NT is very similar, and any difference are explicitly specified.

Once your caching and filtering proxy server has started, you can access it from your Web browser located on the same machine or on another machine wired to the network. Remember to disable caching on your browser, and enter the URL of the server's front page. We typed:

```
http://venus.itso.ral.ibm.com
```

The proxy server's response to this request was similar to the following screen:



Figure 116. Caching and Filtering Component Front Page

This is the home page of the proxy server, named Frntpage.html. To open the Configuration and Administration Forms for the proxy function, click on **Proxy Configuration**. If you have not used the Configuration and Administration Forms since you have started your browser, you will be prompted for the proxy server administrator's User Name and Password, as shown next:

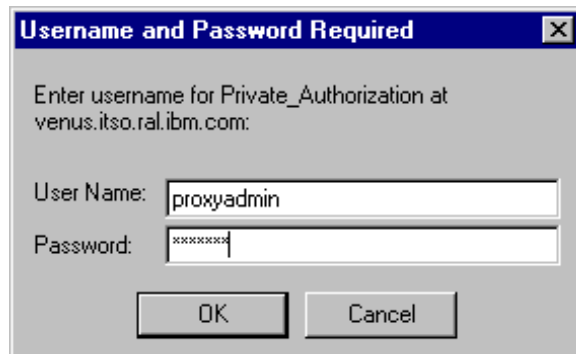


Figure 117. Entering Administrator User Name and Password

Enter the same administrator ID and password you defined during the installation process (see 3.7.1, "Installation on AIX" on page 120 and Figure 110 on page 145), and after the authentication you will get the following screen:



Figure 118. Proxy Configuration and Administration Form

From this menu page, you can select each of the input forms by clicking on the form name.

3.8.1 Using the Configuration and Administration Forms

When opening a form, its input fields and its summary tables are filled out with the current configuration values. The form provides you with a short description about what you can do and how you can change the values. But if you want detailed information for a particular form, you can access the help page by clicking **Help**. We suggest you consult the help, because we found it very useful.

Below the input fields on each form, you find two buttons: Apply and Reset.

After you fill in a particular form, you must click the **Apply** button to update the server configuration with your changes. If you decide not to use the changes you made to the form, click the **Reset** button, and the fields on the form return to the values they had when you first came to the form.

After clicking **Apply**, a message will be displayed indicating whether your input was accepted. If so, you will see a Confirmation page that reports the new setting for the directives. The Confirmation page may contain a **Restart Server** button. You must click it in order to have your changes take effect and the server begin to use the configuration changes. If the Confirmation page does not contain a Restart Server button, then you need to stop your server and start it again (see 3.7.2, “Restarting the Caching and Filtering Proxy Server on AIX” on page 128 and 3.7.4, “Restarting the Caching and Filtering Proxy Server on Windows NT” on page 147).

To restart the caching and filtering proxy server on Windows NT, you can also interact with the GUI (see Figure 115 on page 149). Open the **File** menu and click **Restart**.

3.8.2 Enabling Proxy Function

To enable the proxy function and to indicate which protocols Web Traffic Express should process, select **Proxy Server Settings** from the Proxy Configuration and Administration Forms menu shown in Figure 118 on page 152, and you will get the Proxy Server Settings form, which is similar to the following figure:

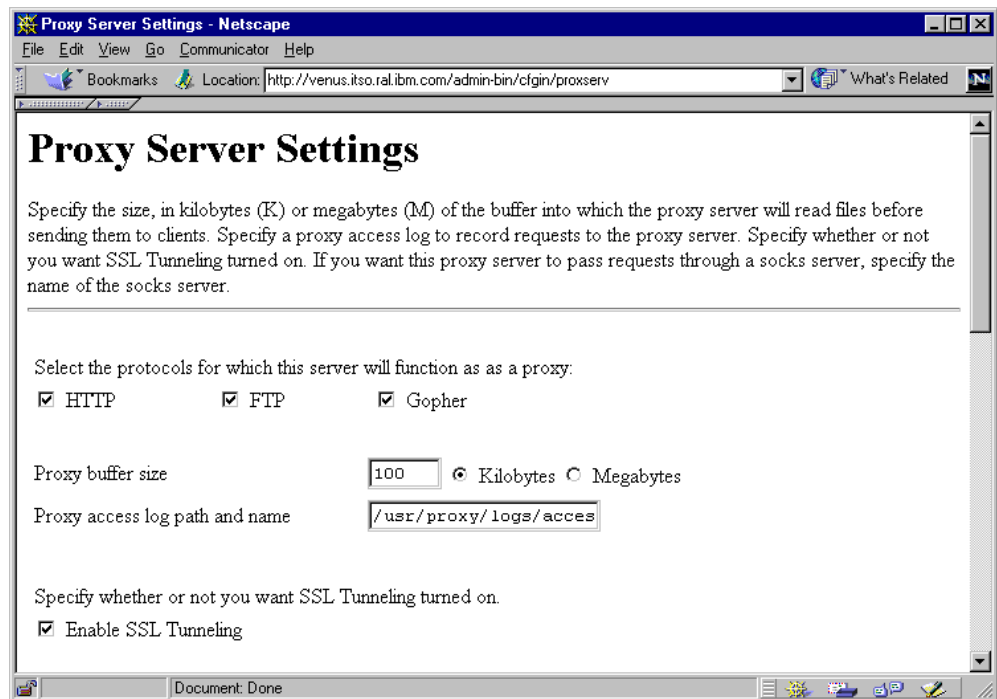


Figure 119. Proxy Server Setting

The initial settings configure the Caching and Filtering component of IBM WebSphere Performance Pack to be a proxy server for the protocols HTTP, FTP and Gopher. We accepted these default settings.

Using the Directives

In the httpd configuration file, use the directive `Proxy` to specify the protocols. The initial settings are:

```
Proxy http:*  
Proxy ftp:*  
Proxy gopher:*
```

If you want to instruct the caching and filtering proxy server to enable SSL tunneling on the TCP well-known port 443 (see 3.6.5, “SSL Tunneling” on page 120), check off **Enable SSL Tunneling** in the Proxy Server Settings form.

Using the Directives

To enable SSL tunneling using the configuration files, ensure that the following two directives are present in the httpd configuration file, where 443 is the port number on the destination server.

```
Proxy *:443  
ENABLE CONNECT
```

In the Proxy Server Settings form you can set the value of Proxy buffer size. When the proxy returns dynamic data (output data from CGI-BIN programs, API programs, server-side includes, Java servlets) it must buffer the data, and you can set the value of the buffer size in the Proxy buffer size field. We accepted the default value, which is 100KB on AIX and 50KB on Windows NT.

Using the Directives

To set the value of Proxy buffer size equal to 100KB use the following directive in the httpd configuration file:

```
MaxContentLengthBuffer 100 K
```

Another parameter that appears in the Proxy Server Settings form is Proxy access log path and name, which allows you to specify the path and file name where the proxy server will log access statistics. Each day at midnight, the caching and filtering proxy server, if it is running, starts a new log file. It uses the specified file name and appends a date suffix as an extension.

Default values on our systems were:

- /usr/lpp/internet/server_root/logs/proxy-log on AIX
- D:\WWW\Log\proxy-log on Windows NT

It is a good idea to remove old log files, since they can take up a significant amount of space on your hard drive. In our situation, on AIX, we even changed the default value and we decided to create a new file system that only stored log files. So we created a new file system whose mount point we named /usr/proxy/logs, and then we entered /usr/proxy/logs/access/httpd-proxy in the Proxy access log path and name field. Then, for example, the log file that the

caching and filtering proxy server creates for September 5, 1998 would be /usr/proxy/logs/access/httpd-proxy.Sep051998.

Using the Directives

To manually specify the path and file name for proxy access log files on AIX, we should have entered the following directive in the httpd.conf configuration file:

Notice, however, that permissions are important while changing the default directory where the log files can exist. The Caching and Filtering component will write to that directory as the user ID/group ID specified in the httpd.conf configuration file (nobody/nobody by default). So if you end up creating your own directories, you must make sure that the caching and filtering proxy server's user ID can write to that directory.

The process of changing the default proxy access log path and file name on Windows NT would be very similar.

For the configuration changes to be written in the configuration files, click **Apply** on the Proxy Server Settings, and you will get the Confirmation page that summarizes the new directives settings:



Figure 120. Confirmation Page

For the time being, only the httpd configuration file has changed. To actually invoke the changes, click the **Restart** button in the Confirmation page (see previous figure), and you will see the Restart Confirmation page, shown in the following figure:

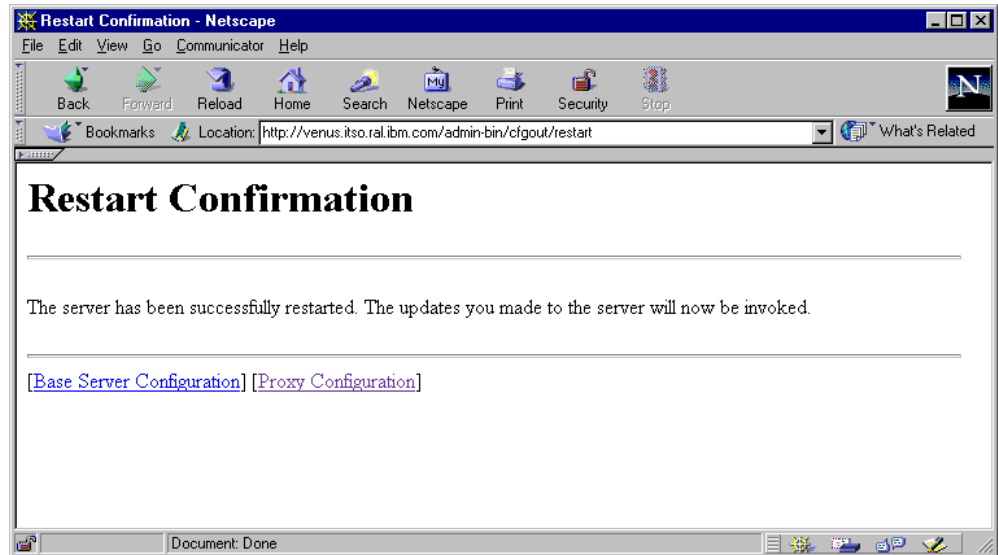


Figure 121. Restart Confirmation Page

From the Confirmation page or the Restart Confirmation page, click **Proxy Configuration** to come back to the Proxy Configuration and Administration Forms, as shown in Figure 118 on page 152.

3.8.3 Enabling the Pure Proxy Functionality

The Caching and Filtering component of IBM WebSphere Performance Pack can act as both a content server and a proxy server. For best performance, you are suggested to use it only as a proxy server. In fact, by default, the initial settings instruct this component to act as a pure proxy server. You can see this if you select **Proxy Performance** from the initial menu, shown in Figure 118 on page 152. You will get the following form:

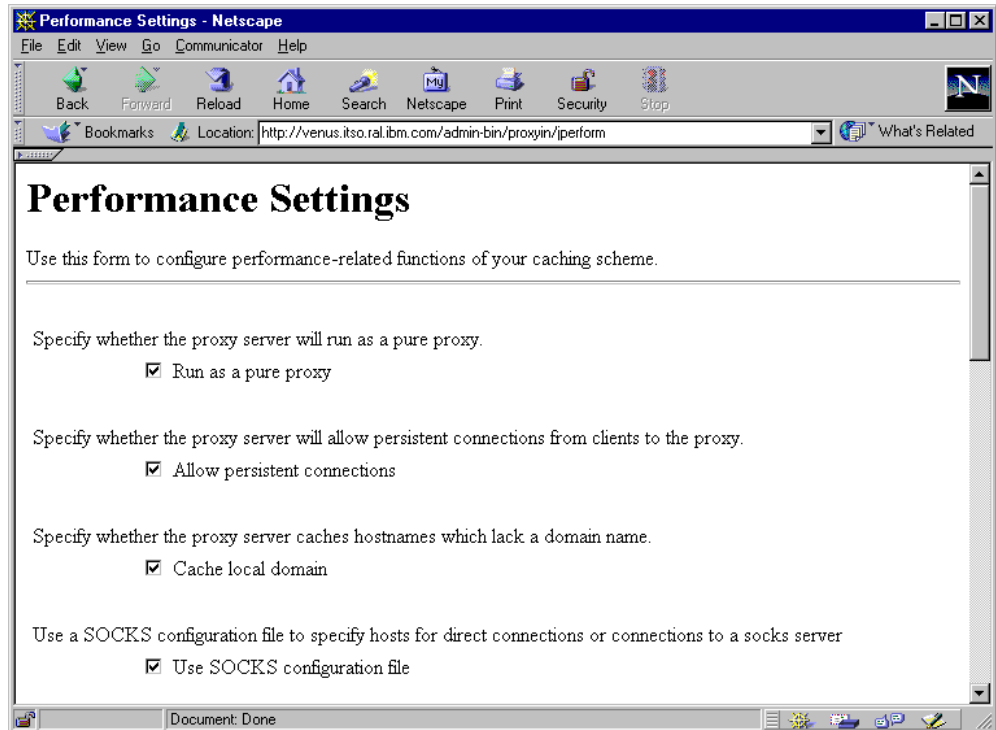


Figure 122. Performance Settings

The parameter we were interested in was Run as pure proxy. We kept the default configuration, with the **Run as a pure proxy** check box marked, because we wanted to use the Caching and Filtering component as a pure caching and filtering proxy server.

Using the Directives

The directive for setting the role of the Caching and Filtering component of IBM WebSphere performance Pack is in the javelin configuration file:

```
PureProxy On.
```

We suggest you verify that your caching and filtering proxy server is configured to act as a pure proxy server and not as a content server. After our verification, we did not need to change anything in the above form, so it was not necessary to click the **Restart** button.

3.8.4 Enabling Basic Caching

Now we show you how to enable the basic caching capabilities of the Caching and Filtering component of IBM WebSphere Performance Pack. In this way the caching and filtering proxy server can save copies of the files that the clients request, and can serve them quickly from its own cache when they are requested again. Notice however that not all files can be cached. We showed a list of non-cacheable files in 3.3.1, "Controlling Which Documents Are Kept in the Cache" on page 110.

For turning caching on, select **Caching Settings** from the Proxy Configuration and Administration Forms home page (see Figure 118 on page 152). The Caching Settings form, shown in the following figure, will be displayed:

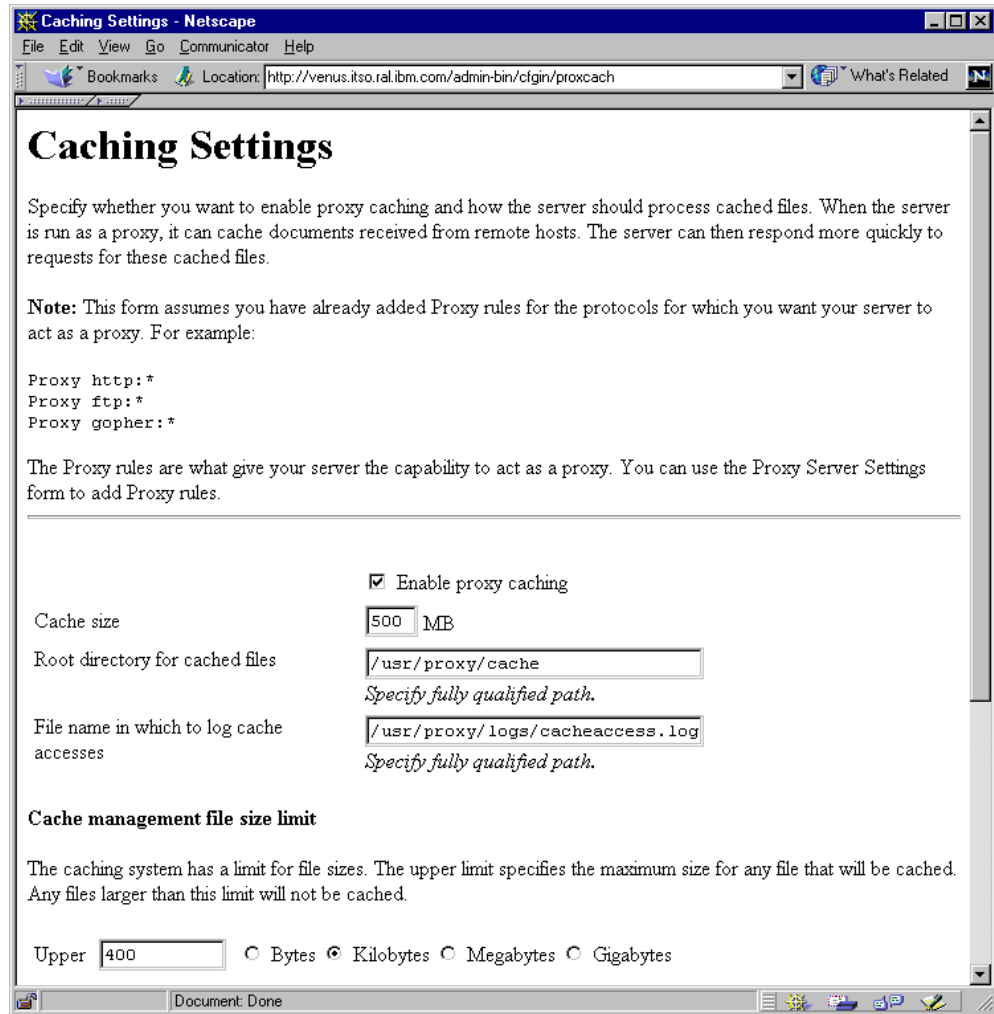


Figure 123. Caching Setting Form

Check on the **Enable proxy caching** check box to enable the caching capabilities.

Use the Cache size field to enter the maximum amount (in MB) of disk space you want to reserve to the proxy cache. We accepted the default value of 500MB, but if you have at your disposal plenty of disk space, you can increase the default value.

Using the Directives

To manually set to 500MB the disk space size reserved to the cache, use the directive found in the httpd configuration file.

```
CacheSize 500
```

In the text field named Root directory for cached files enter the directory under which all files will be cached. Notice that no value is defined by default. In our situation, on the AIX platform, we decided to create a new file system whose mount point we named `/usr/proxy/cache`, and then we entered `/usr/proxy/cache` in the Root directory field.

Using the Directives

To manually specify the path of the cache root directory as `/usr/proxy/cache` on the AIX platform, enter the following directive in the httpd.conf configuration file:

```
CacheRoot /usr/proxy/cache
```

Furthermore, in the text field named File name in which to log cache accesses, you can specify the path and file name where you want the server to store a daily log of its cache accesses. For example, on our AIX platform, we entered `/usr/proxy/logs/cacheaccess.log`.

Using the Directives

To manually specify the path and file name in which to log cache accesses as `/usr/proxy/logs/cacheaccesses.log`, we should have entered the following directive in the httpd.conf configuration file on our AIX platform:

```
CacheAccessLog /usr/proxy/logs/cacheaccess.log
```

We want to remind you here of the same recommendation we mentioned in 3.8.2, “Enabling Proxy Function” on page 153. You must be careful with the permissions and make sure that the caching and filtering proxy server’s user (on AIX, by default, nobody/nobody) can write to the directories where the cache is going to be.

At last, you can also specify the maximum size of the files to be cached, by entering a value in the Upper text field. The value can be specified in Bytes, Kilobytes, Megabytes or Gigabytes. Files larger than the specified size will not be cached. We accepted the default value of 400KB.

Using the Directives

To specify the maximum size for any file that will be cached as 400KB, enter the following directive in the httpd configuration file, according to the convention that B means bytes, K means kilobytes, M means megabytes and G means gigabytes:

```
CacheLimit_2 400 K
```

If you used the Configuration and Administration Forms rather than the configuration files and you want the configuration changes to be written in the configuration files, click **Apply** on the Caching Settings form, and you will get the Confirmation page that summarizes the new directives settings:

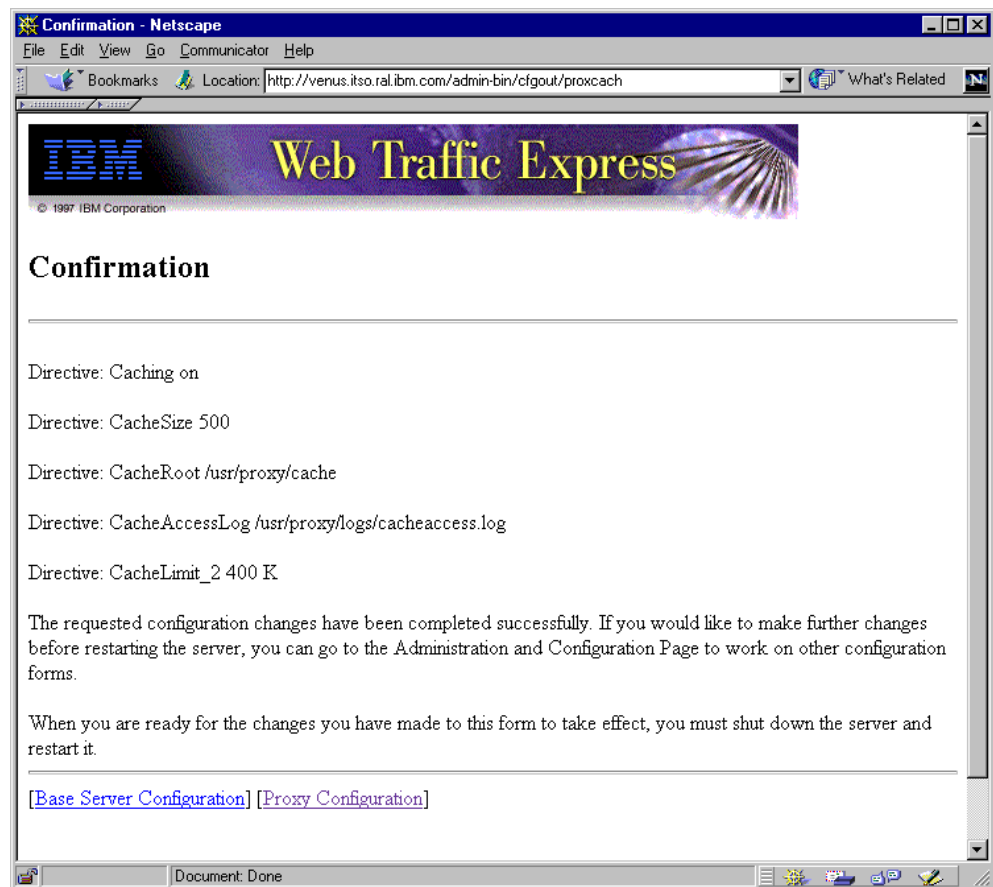


Figure 124. Confirmation Window

In this case, in order to have the changes take effect, you must stop and restart the server (refer to 3.7.2, “Restarting the Caching and Filtering Proxy Server on AIX” on page 128 and 3.7.4, “Restarting the Caching and Filtering Proxy Server on Windows NT” on page 147).

Caching Problems?

After stopping and restarting the proxy server, we realized that the proxy server was not caching any files. This happened because on AIX the proxy runs, by default, as user nobody. Such a user did not have write access to the directory /usr/proxy/cache we had specified as root directory for cached files. So we gave all access to the user nobody by changing the user and group who owned the root directory for cached files. From a command line, when the current directory was /usr/proxy, we entered the following command:

```
chown nobody:nobody cache
```

3.8.5 Garbage Collection

Once you have enabled caching, you need your server to periodically perform the so-called *garbage collection* process (see 3.3.3, “Cache Size and Garbage Collection” on page 114). In this way you can prevent the cache of your proxy server from growing beyond the maximum size that you set for it (see 3.8.4, “Enabling Basic Caching” on page 158). The garbage collection process deletes all the expired files, which are the cached files that should no longer be cached. By default, the garbage collection process is enabled, performed once a day, at 3:00 a.m., and allocates 1000KB of RAM memory.

We changed only the last of the above values, and set the maximum RAM memory that the garbage collection process can allocate to 2000KB. In fact The garbage collection process is best performed if there is enough memory.

To see the garbage collection configuration, select **Cache Storage Reuse** from the Proxy Configuration and Administration Forms home page, shown in Figure 118 on page 152, and you will get the Cache Storage Reuse form, which should be similar to the following figure:

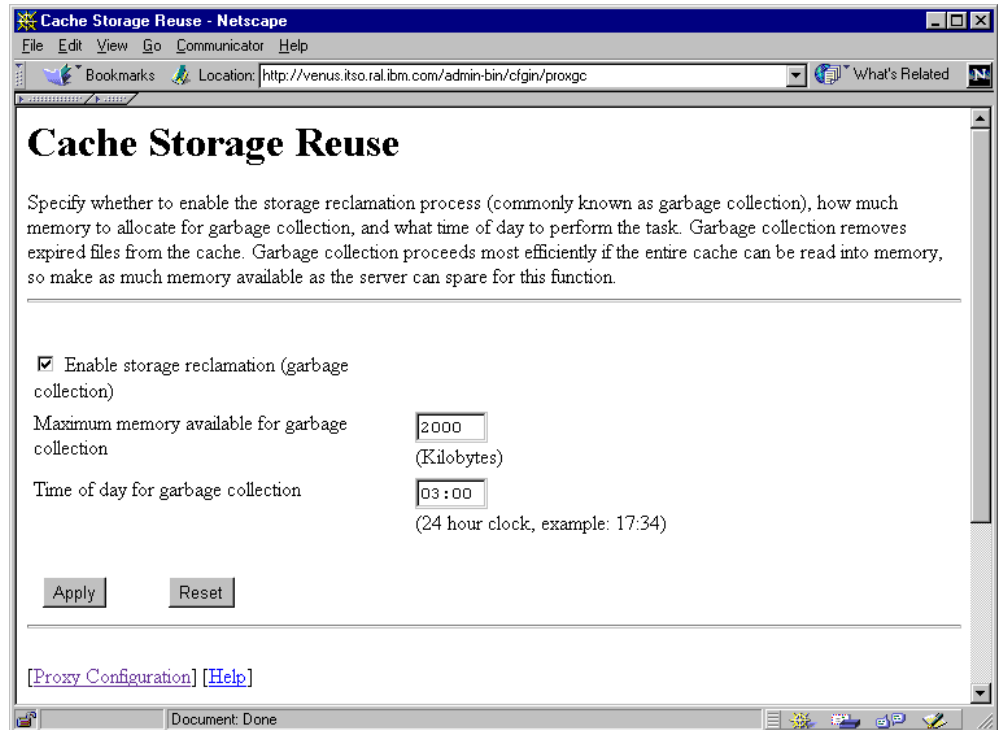


Figure 125. Cache Storage Reuse

Notice that the time value must be specified in 24 hour clock format.

The garbage collection process makes use of a lot of machine resources, such as CPU and RAM memory. For this reason, it is suggested to run the garbage collection when the load of requests is estimated as low as possible.

Using the Directives

To manually enable garbage collection, the right directive would be:

```
Gc On
```

To set to 3:00 a.m. the time at which garbage collection is performed, use the directive:

```
GcDailyGC 03:00
```

To manually set the maximum memory available for garbage collection to 2000KB, the right directive is

```
GcMemUsage 2000
```

All the above directives must be entered in the httpd configuration file.

In order for the configuration changes to be written in the configuration files, click **Apply** on the Cache Storage Reuse form and you will get the Confirmation page, which summarizes the new directives settings, as shown next:

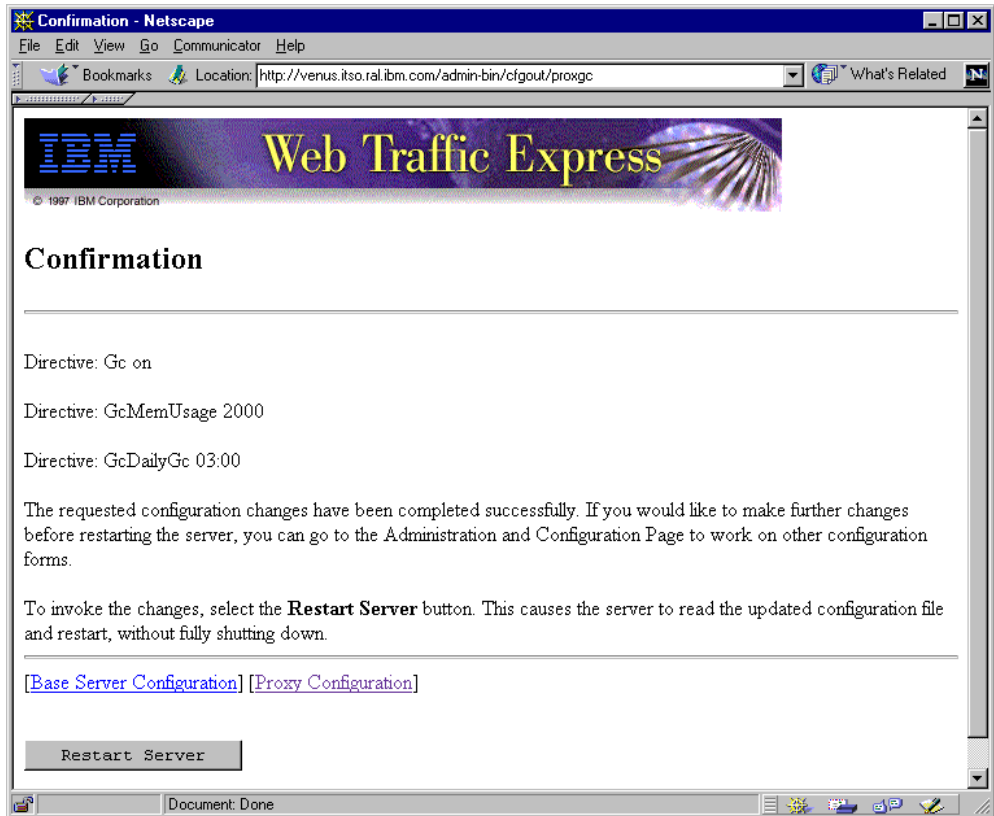


Figure 126. Cache Storage Reuse Confirmation Page

To actually have your changes take effect, click the **Restart** button in the above Confirmation page, and you will see the Restart Confirmation page (see Figure 120 on page 156).

3.9 Proxy Chaining Scenario

Proxy chaining is a mechanism that allows you to create a hierarchical chain of proxies. A client can send its request to a proxy server that redirects the request to another proxy at a higher level. This one, in turn, can send the request to a higher level proxy in the chain, and so forth. This architecture is also known as *daisy chain of proxies* and allows the creation of hierarchies of proxies, each proxy having a certain level in the hierarchy. The last proxy in the chain may direct the request to the origin Web server.

Before forwarding the request to a higher level proxy, each proxy in the chain checks its own cache to see whether the requested file has already been requested and cached and whether it is still valid. If this check fails, the request is forwarded to the proxy that in the hierarchy occupies the higher level. If the file is found in the cache, then it is sent back in the chain.

The following figure shows a graphical representation of the proxy chaining process:

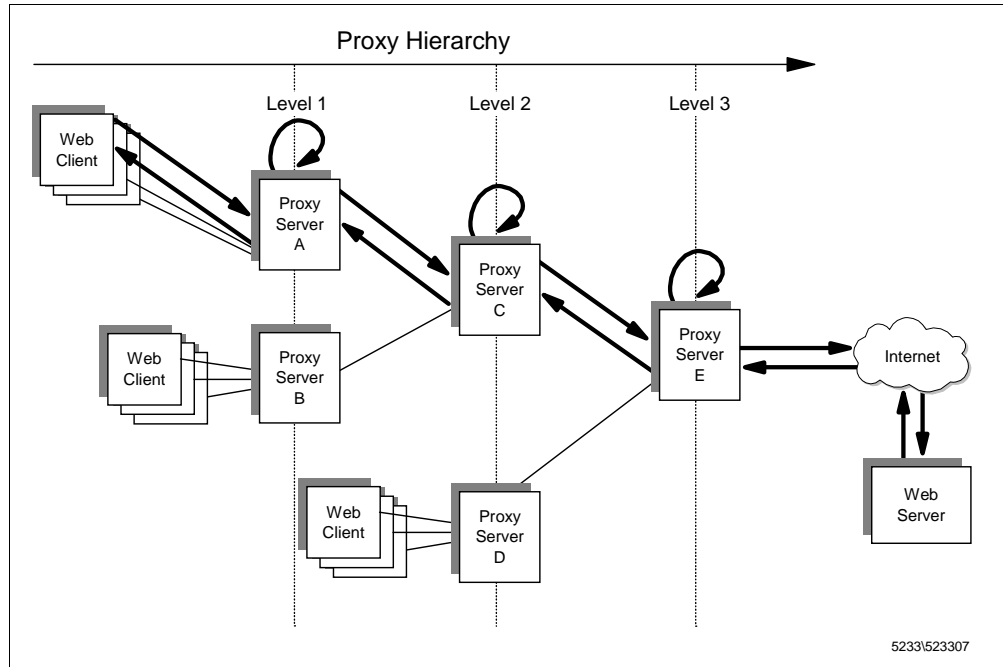


Figure 127. Graphical Representation of the Proxy Chaining

Notice that the request goes through the daisy chain of servers and the response passes through the same proxies but in the reverse way. If intermediate proxies have caching enabled, each of them would search its own cache for the requested resource and return a cached copy, without forwarding the request to the origin Web server.

Advantages

- Proxies at a lower level, which are closer to the client that originated the request, benefit from the caches of the higher level proxies.
- Proxy chaining reduces the load on the highest level proxy (typically, the proxy closest to the firewall) and ultimately on the content server, since lower level proxies may already have the document cached.
- The larger the number of users, the higher the probability that the proxy server already has the document in its cache. Considering that high-level proxies serve a larger number of clients, many requests that cannot be honored by a low-level proxy can be resolved by higher level proxies, since other groups of clients may have already requested the same files.

Disadvantages

- A high-level proxy should have a larger cache, since it has to honor the requests of a large number of users.
- Proxy chaining greatly increases response time for requests, especially for those files that have not been cached yet by any proxies in the hierarchy.
- The risk of failure in a chain increases with each additional node.

The Caching and Filtering component of IBM WebSphere Performance Pack allows the creation of proxy chains based on the protocol (HTTP, FTP or Gopher) of the request to be forwarded. In other words, a proxy server can be configured to send all incoming requests with a particular protocol to a higher level proxy server in the chain.

3.9.1 How to Set Up a Proxy Chaining Environment

The following figure shows the architecture of the environment where we performed proxy chaining:

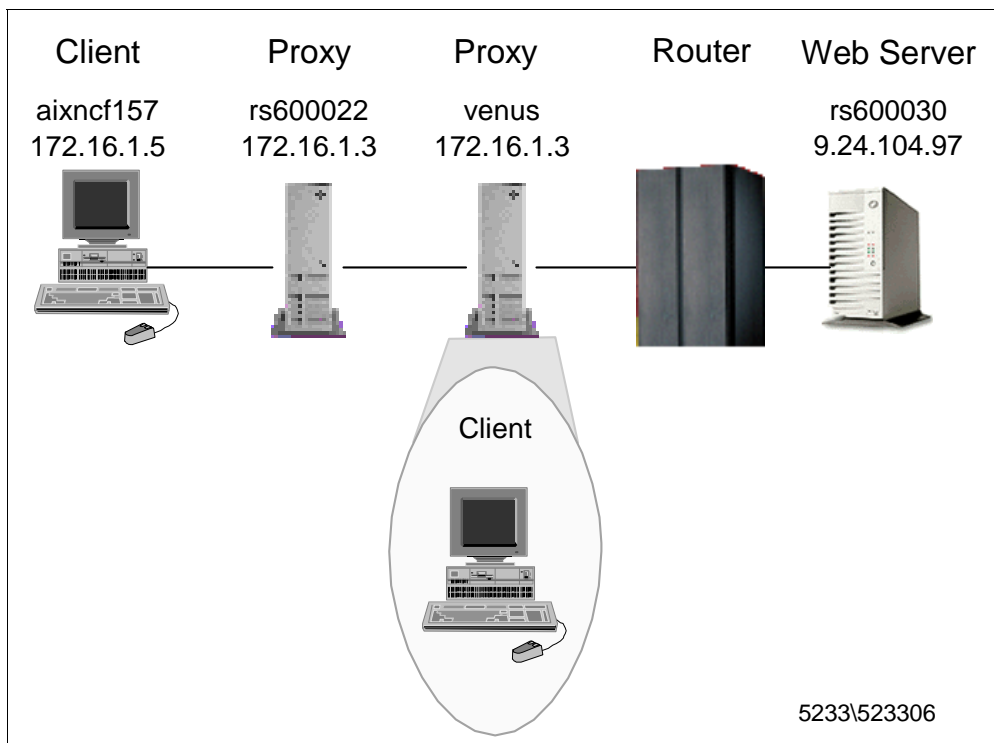


Figure 128. Architecture of the Proxy Chaining Environment

We had two caching and filtering proxy servers belonging to the chain. The first level proxy (installed on an AIX machine whose host name was rs600022 and IP address was 172.16.1.2) was configured to forward the requests its cache could not satisfy to the second level proxy (installed on another AIX machine, whose host name was venus and IP address was 172.16.1.3).

Both the proxy servers had the cache enabled.

On the machine rs600022, we defined:

- The root directory for cached files as /usr/proxy/cache
- The cache access log file as /usr/proxy/logs/rs600022-cache

On the machine venus, we defined:

- The root directory for cached files as /usr/proxy/cache
- The cache access log file as /usr/proxy/logs/venus-cache

As you can see in the above figure, we had a Web browser installed on a client machine, whose host name was aixncf157 and IP address was 172.16.1.5. This Web browser was configured to use rs600022 as the proxy server. This Web browser played the role of a first-level client, since it was configured to forward its own requests to a first-level proxy, installed on rs600022.

Another Web browser was installed on the same machine venus, having IP address 172.16.1.3. This Web browser has the role of the second-level client machine, since it was configured to forward its own requests to the second-level proxy, installed on venus.

As you can see, the first-level proxy in the architecture we had implemented was to serve one single client, while the second-level proxy would receive requests from two clients.

If a file requested by the first-level client was not found in the cache, the request would be directed to the Web content server, placed behind the router. The Web server machine was a RISC/6000 where we had installed AIX 4.3 as the operating system and Lotus Domino Go Webserver 4.6.2.5 as the Web server. The host name of this machine was rs600030 and its IP address was 9.24.104.97. On this Web server, the httpd log file was /usr/lpp/internet/server_root/logs/httpd-log.Oct041998.

The following table summarizes the configuration of the environment where we performed our experience:

Table 2. Environment Configuration for Proxy Chaining

Role	Operating System	Host Name	IP Address
First-Level Client	AIX 4.3.1	aixncf157	172.16.1.5
First-Level Proxy	AIX 4.3.1	rs600022	172.16.1.2
Second-Level Client	AIX 4.3.1	venus	172.16.1.3
Second-Level Proxy	AIX 4.3.1	venus	172.16.1.3
Web Server	AIX 4.3	rs600030	9.24.104.97

To configure proxy chaining, we opened the Proxy Configuration and Administration Forms home page for the proxy server rs600022 and then we selected **Proxy chaining and Non-Proxy Domains** (see Figure 118 on page 152). On doing so, we got the Proxy Chaining and Non-Proxy Domains form, as shown next:

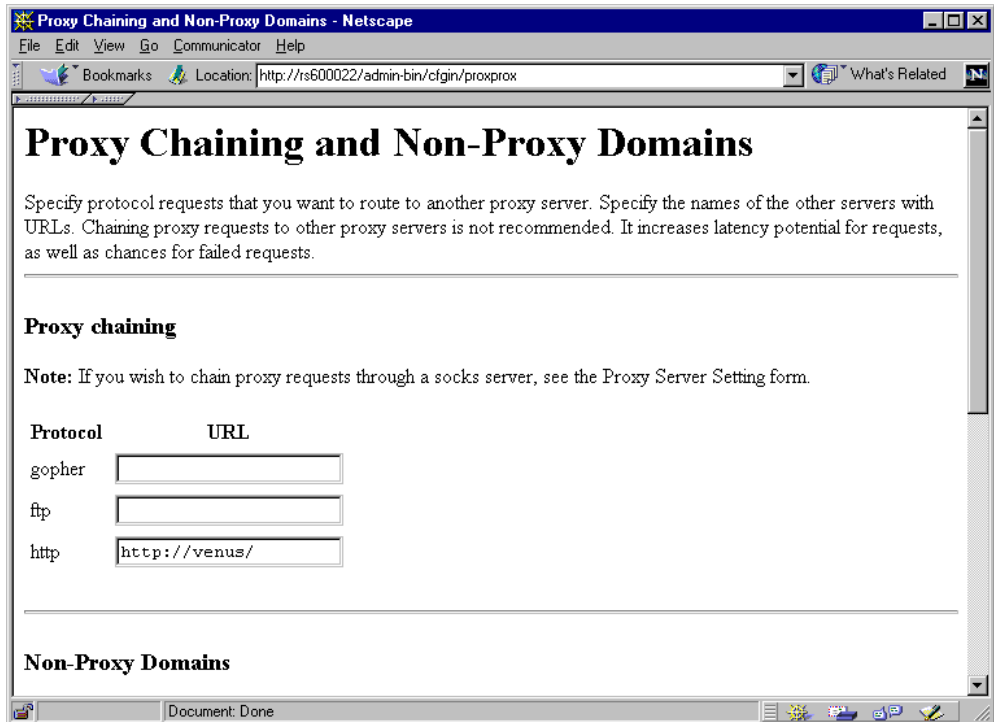


Figure 129. Proxy Chaining and Non-Proxy Domains Form

We wanted this proxy to forward all HTTP requests to the proxy venus, so we entered in the http text field the URL:

```
http://venus/
```

Notice that the training slash must be included in the URL.

Using the Directives

To specify the venus as the higher level proxy in the chain, we could also have entered the following directive in the httpd configuration file:

```
Http_Proxy http://venus/
```

Then we clicked **Apply** to update the first-level proxy server rs600022 with the changes. We got the following Confirmation panel:



Figure 130. Proxy Chaining Confirmation Page

This confirmed that the configuration files had been changed according to our settings. To actually have our changes take effect, we selected the **Restart Server** button in the Confirmation page.

Notice that proxy chaining is not enabled by default and you have to manually configure all the proxy servers that need to recognize a higher level proxy server.

3.9.2 Proxy Chaining Experiences

After the steps described in 3.9.1, “How to Set Up a Proxy Chaining Environment” on page 166, we wanted to verify that our proxy chaining implementation worked correctly.

We stopped both the proxy servers by issuing on both machines the command:

```
stopsrc -s httpd
```

To better follow the test, we cleaned the logs and caches on both machines, by removing all files and subdirectories under the directories `/usr/proxy/cache` and `/usr/proxy/logs`.

After doing this, we restarted both the proxy servers with the command:

```
startsrc -s httpd
```

We also stopped the `httpd` daemon on the Web server machine, then deleted the `httpd` log file `/usr/lpp/internet/server_root/logs/httpd-log.Oct041998`, and finally restarted the `httpd` daemon.

Then we wanted to follow the online growth of the log files in consequence of our actions. Here is what we did:

1. On rs600022, when the current directory was /usr/proxy/logs, we entered the following command:

```
tail -f rs600022-cache.Oct041998
```

2. On venus, when the current directory was /usr/proxy/logs we entered the following command:

```
tail -f venus-cache.Oct041998
```

3. On rs600030, when the current directory was /usr/lpp/internet/server_root/logs, we entered the following command:

```
tail -f httpd-log.Oct041998
```

3.9.2.1 First Experience

From the Web browser installed on the second-level client machine venus, with the HTTP second-level proxy set to venus, we requested the URL <http://rs600030/afs/webstone/html/file8k.html>.

What happened to our request?

1. The request from the Web browser was taken by the proxy server venus.
2. The proxy server venus looked up its cache and did not find the file, so no access was done to the venus proxy server cache:

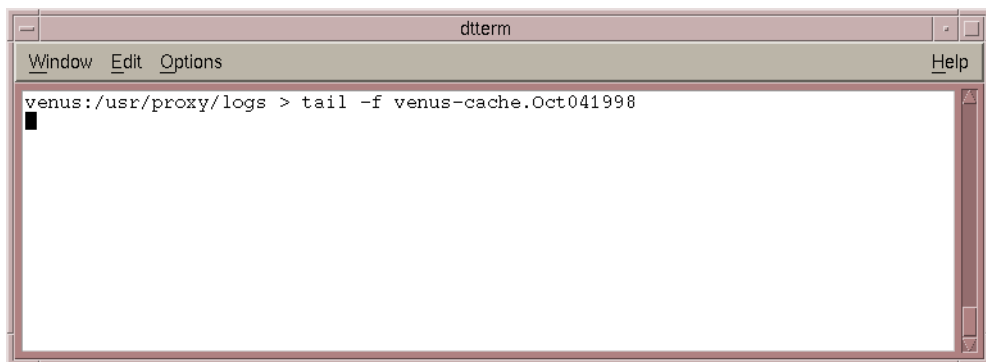


Figure 131. venus Cache Access Log File - 1

3. The proxy server venus requested the file to the Web server.
4. The Web server registered the access and gave back the page file8k.html to the proxy server venus, whose IP address was 172.16.1.3:

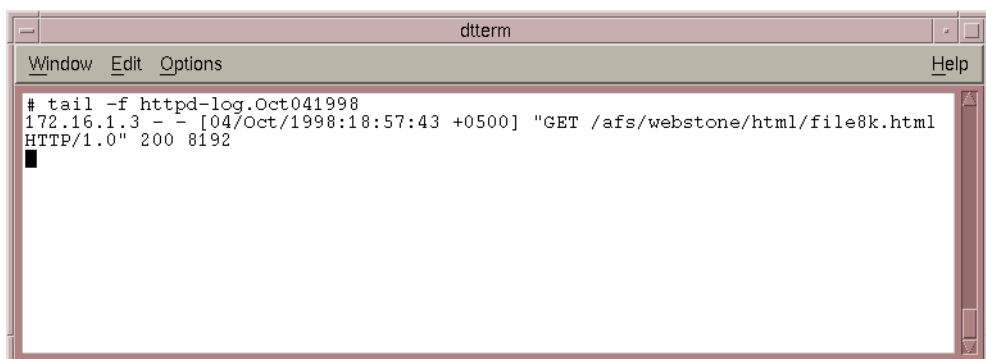


Figure 132. rs600030 Httpd Log File - 1

5. The proxy server venus cached the page and sent it to the Web browser.

Obviously, no access has been done to the rs600022 proxy server cache, as shown in the following figure:

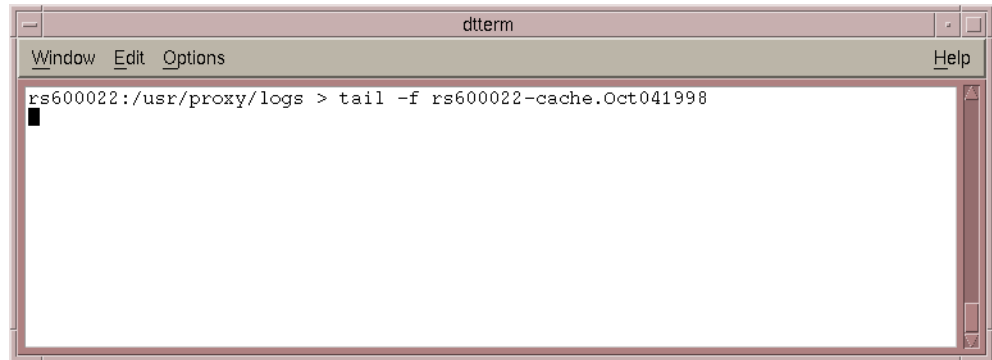


Figure 133. rs600022 Cache Access Log File - 1

So at this point the file file8k.html has been cached by venus, but not by rs600022.

3.9.2.2 Second Experience

From the Web browser installed on the first-level client machine aixncf157, with HTTP first-level proxy set to rs600022, we requested the URL: <http://rs600030/afs/webstone/html/file8k.html>.

What happened this time to our request?

1. The request from the Web browser was taken by the proxy server rs600022.
2. The proxy server rs600022 looked up its cache and did not find the file, so no access was done to the rs600022 proxy server cache:

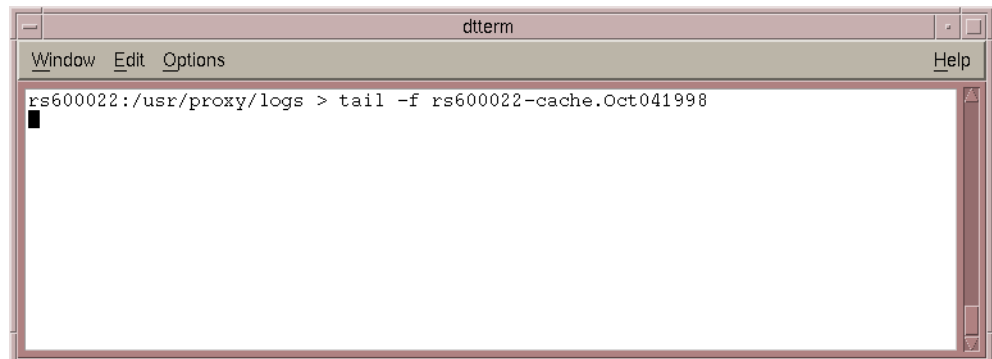


Figure 134. rs600022 Cache Access Log File - 2

3. The proxy server rs600022 requested the file to the higher level proxy server in the chain, which was venus. In turn, venus looked up its cache, where it found the file file8k.html, and sent it back to the proxy rs600022. Then, in this case, the proxy rs600022, having IP address 172.16.1.2, accessed the cache of venus, as reported in the following figure:

```
dtterm
Window Edit Options Help
venus:/usr/proxy/logs > tail -f venus-cache.Oct041998
172.16.1.2 - - [04/Oct/1998:18:58:36 +0500] "GET http://rs600030/afs/webstone/html/file8k.html HTTP/1.0" 200 8192
█
```

Figure 135. venus Cache Access Log File - 2

4. The proxy server rs600022 cached the file file8k.html, and gave it back to the Web browser.

Obviously, no access had been done to the Web server, as shown from the following figure:

```
dtterm
Window Edit Options Help
# tail -f httpd-log.Oct041998
172.16.1.3 - - [04/Oct/1998:18:57:43 +0500] "GET /afs/webstone/html/file8k.html HTTP/1.0" 200 8192
█
```

Figure 136. rs600030 Httpd Log File - 2

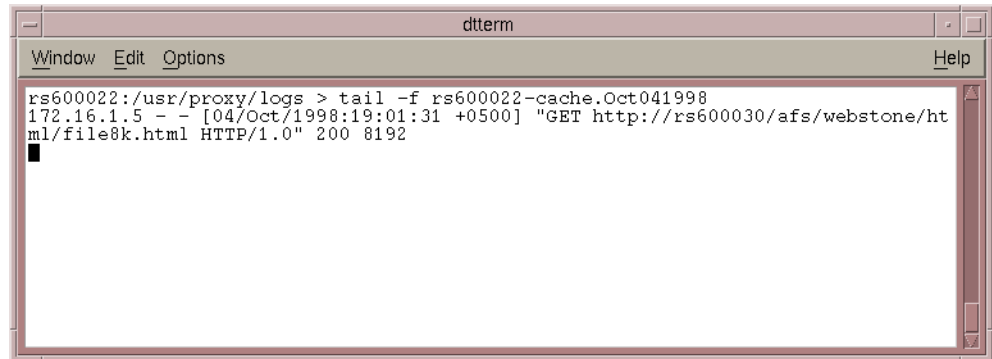
So at this point the file file8k.html has been cached by both the proxy servers venus and rs600022.

3.9.2.3 Third Experience

From the Web browser installed on the first-level client aixncf157, with HTTP first-level proxy set to rs600022, we requested again the URL `http://rs600030/afs/webstone/html/file8k.html`.

What happened this time to our request?

1. The request coming from the Web browser was taken by the proxy server rs600022.
2. The proxy server rs600022 looked up its cache, this time found the file file8k.html and give it back to the Web browser. Then, in this case, the first-level client machine, having IP address 172.16.1.5, accessed the cache of rs600022, as reported in the following figure:



```
dtterm
Window Edit Options Help
rs600022:/usr/proxy/logs > tail -f rs600022-cache.Oct041998
172.16.1.5 - - [04/Oct/1998:19:01:31 +0500] "GET http://rs600030/afs/webstone/html/file8k.html HTTP/1.0" 200 8192
█
```

Figure 137. rs600022 Cache Access Log File - 3

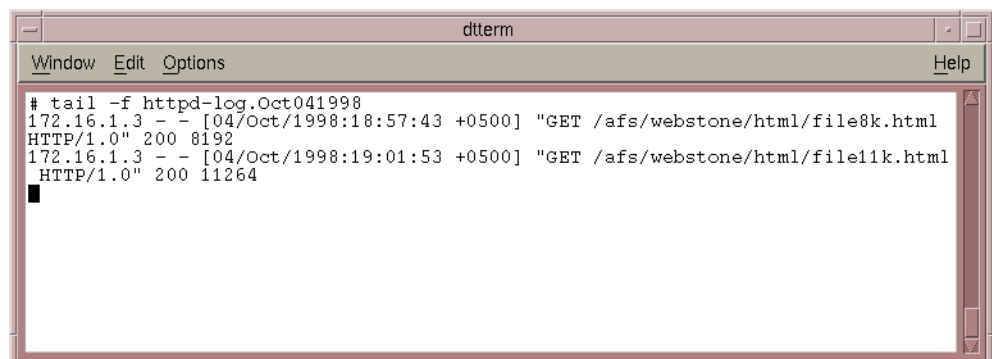
Obviously, this time the caches of venus and rs600030 were not addressed.

3.9.2.4 Fourth Experience

Now, from the Web browser installed on the first-level client machine aixncf157, with HTTP first-level proxy set to rs600022, we requested the URL `http://rs600030/afs/webstone/html/file11k.html`, which had never been requested before.

What happened this time to our request?

1. The request coming from the Web browser was taken by the proxy server rs600022.
2. The proxy server rs600022 looked up its cache and did not find the file, so the rs600022 proxy server cache was not accessed.
3. The proxy server rs600022 requested the file to the second-level proxy server in the chain, which was venus.
4. Proxy venus looked up its cache. It did not find the file file11k.html.
5. The proxy server venus requested the file to the Web server.
6. The Web server registered the access and gave the page file11k.html back to the proxy server venus. So in this case the proxy server venus, having IP address 172.16.1.3, accessed the Web server, as shown in the following figure:



```
dtterm
Window Edit Options Help
# tail -f httpd-log.Oct041998
172.16.1.3 - - [04/Oct/1998:18:57:43 +0500] "GET /afs/webstone/html/file8k.html
HTTP/1.0" 200 8192
172.16.1.3 - - [04/Oct/1998:19:01:53 +0500] "GET /afs/webstone/html/file11k.html
HTTP/1.0" 200 11264
█
```

Figure 138. rs600030 Httpd Log File - 4

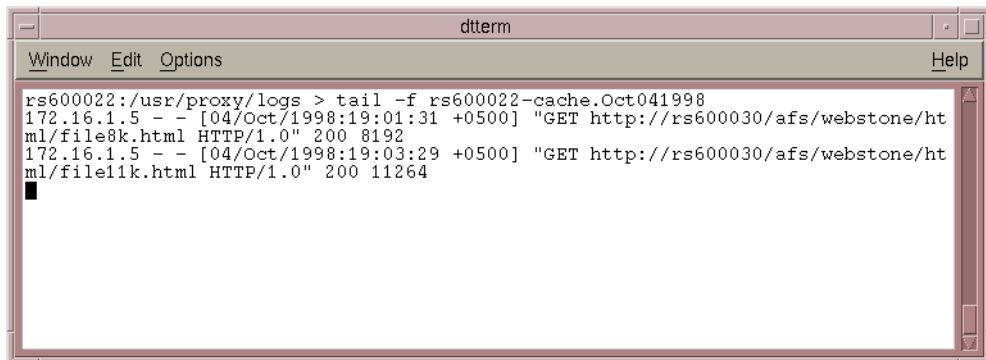
7. The proxy server venus cached the page and sent it to the first-level proxy server rs600022.
8. The proxy rs600022 cached the file file11k.html and gave it back to the Web browser aixncf157 that first had requested the page.

3.9.2.5 Fifth Experience

From the Web browser installed on the first-level client aixncf157, with HTTP first-level proxy set to rs600022, we requested again the URL `http://rs600030/afs/webstone/html/file11k.html`.

What happened this time to our request?

1. The request coming from the Web browser was taken by the proxy server rs600022.
2. The proxy server rs600022 looked up its cache, this time found the file file11k.html this time and could give it back to the Web browser. Then, in this case, the Web browser installed on aixncf157, with IP address 172.16.1.5, accessed the cache of rs600022, as reported in the following figure:



```
rs600022:/usr/proxy/logs > tail -f rs600022-cache.Oct041998
172.16.1.5 - - [04/Oct/1998:19:01:31 +0500] "GET http://rs600030/afs/webstone/html/file8k.html HTTP/1.0" 200 8192
172.16.1.5 - - [04/Oct/1998:19:03:29 +0500] "GET http://rs600030/afs/webstone/html/file11k.html HTTP/1.0" 200 11264
```

Figure 139. rs600022 Cache Access Log File - 5

Obviously, no access had been done to the caches of venus or rs600030.

3.10 Proxy Server Protection

In this section we describe how to control who can access the proxy function of the Caching and Filtering component of IBM WebSphere Performance pack. It is possible for you to specify, for example, that only authorized users can access your caching and filtering proxy server, by requiring them to authenticate with user ID and password. As another example, you can restrict access to your caching and filtering proxy server, allowing only requests generated by users belonging to specified domains to come.

You can perform the appropriate configuration by editing the httpd configuration file or by using the Configuration and Administration Forms. In our experience, we directly modified the httpd configuration file, since in this case we considered this procedure more useful to understand the underlying directives.

The machine where we had installed the Caching and Filtering component was a RISC/6000, with AIX 4.3.1 as the operating system. The machine host name was venus and its IP address was 172.16.1.3.

3.10.1 Resource Protection

In general, two directives are necessary in the httpd configuration file to protect resources: `Protect` and `Protection`.

Using the `Protection` directive, you can specify a set of rules that compose a particular protection strategy. Then you are required to specify the resources to be protected, and you can do that by using the `Protect` directive. The `Protect` directive allows you to define the resources you want to protect and the name of the protection strategy you want to use.

So all you need to do is define in the httpd configuration file your own protection setup through the `Protection` directive, and then use the `Protect` directive to apply that protection setup to the proxy function.

A protection setup is defined using the `Protection` directive. The `Protection` directive has the following form, where `label-name` is whatever you want to name your protection setup (for example, `PROXY-PROT`), and each `subdirective value` entry defines how you want the resources that will use this protection setup to be accessed:

```
Protection label-name {  
    subdirective value  
    subdirective value  
    ...  
}
```

To define the resources to be protected, you should use the `Protect` directive, which has the following form, where `resource` is the resource you want to protect, and `label-name-of-Protection` is the label name of the protection setup you want to use:

```
Protect resource label-name-of-Protection
```

Note that the `Protection` directive must be placed before any `Protect` directive that points to it; otherwise the caching and filtering proxy server would not be able to associate any protection setup to the label `label-name-of-Protection`.

It is also necessary to put all the `Protect` directives before any `Pass` or `Exec` directives in the httpd configuration file.

3.10.2 Directives to Protect Access to the Proxy

In the `httpd.conf` file you can find two particular examples of protection setups: `PROXY-PROT` and `PROT-ADMIN`. What you would need to do is define your own protection setup, using either (or both) of these existing setups as examples. We modified the existing `PROXY-PROT` example, and wrote the following protection setup to restrict access to the proxy by requiring a user ID and password to all the users:

```

Protection PROXY-PROT {
    ServerId      ProxyServer
    AuthType      Basic
    GetMask       All@(*)
    PutMask       All@(*)
    PostMask      All@(*)
    Mask          All@(*)
    PasswdFile    /usr/lpp/internet/server_root/protect/proxyUsers.pwd
}

Protect http:* PROXY-PROT

```

Below we give an explanation of each line of the above protection setup.

The `Protection` directive defines a protection setup named `PROXY-PROT`.

The `ServerID` subdirective specifies the name, `ProxyServer`, you will use to identify the protection setup to requesters. The name does not need to be a real machine name. When the proxy server sends a requester a prompt for a user ID and password, it will also include the name you specify as the value of the `ServerID` subdirective. Most browsers display this name on the prompt. In this way, the requester is capable of understanding which proxy server sent that prompt, and can decide the user ID and password to send back.

The `AuthType` subdirective specifies the type of authentication. We used the value `Basic` to specify basic authentication, meaning that users are authenticated through user ID and password. In basic authentication, user ID and password are usually encoded in base64 format, but they are not encrypted. (For further details, see the IBM redbook *Internet Security in the Network Computing Framework*, SG24-5220.)

The `PasswdFile` subdirective specifies the path and name of the password file to be used. In our case, the value of this subdirective was:

```
/usr/lpp/internet/server_root/protect/proxyUsers.pwd
```

A password file contains a list of user IDs and passwords. Each user ID has one valid password defined for it. Note that the user ID in the password file does not have any relation to the addresses or host name machines of the requester, nor with the users defined on the underlying operating systems. A password file is created in the proxy server machine itself. To create and maintain password files, you can use the Configuration and Administration Forms or use the `htadm` command. We chose to use the `htadm` command, as we show in 3.10.3, “Using the `htadm` Command to Manage Password Files” on page 178.

The `Mask` subdirectives allow you to specify how one or more of the HTTP methods are to be accessed. For example, you can specify different access levels for `GET` and `POST` methods using the `GetMask` and `PostMask` subdirectives, or prohibit other methods such as `PUT` or `DELETE` from being accessed.

By modifying the values of the various `Mask` subdirectives, you can specify that access is via user ID and password, IP address, or a combination of both.

For example:

- `GetMask All` would specify that any user in the password file specified by the `PasswdFile` subdirective could issue a `GET` request. The user will then be prompted to enter a valid user ID and password as defined in the password file.
- `GetMask All@96.*.*.*` would permit access to all users in the password file, but only if the request came in from an IP address starting with:

96.

Any other requests would be rejected.

Note that `GetMask All@(*)` is equivalent to `GetMask All`.

The `Mask` subdirective allows you to authorize any other enabled method not covered by the other `Mask` subdirectives, such as `GetMask`, `PostMask` and `PutMask`. Notice that these other `Mask` subdirectives take precedence over the `Mask` subdirective if they are all present in the protection setup. The consequence of these considerations is that we could substitute the protection setup we have just shown with the following one:

```
Protection PROXY-PROT {
    ServerId      ProxyServer
    AuthType      Basic
    Mask          All@(*)
    PasswdFile    /usr/lpp/internet/server_root/protect/proxyUsers.pwd
}

Protect http:* PROXY-PROT
```

We preferred to use the first one in order to show you more details about the `Mask` subdirectives.

As you can see, the `Protection` directive and its subdirectives are very flexible and will allow you to define your resource access policies in several different ways or combinations.

All you need to do after defining the protection setup is to use the protection setup in a `Protect` directive to associate it with your proxy resources. We used the following one:

```
Protect http:* PROXY-PROT
```

So in our case all client requests starting with `http:` will cause the caching and filtering proxy server to prompt the user for a user ID and password.

Note that in this situation, client requests that ask to use protocols other than HTTP, such as FTP, would not be protected by the caching and filtering proxy server, since they would start with `ftp:`. To protect this type of request, you should add the following directive:

```
Protect ftp:* PROXY-PROT
```

If you want protect the access to all the caching and filtering proxy server functions using the protection strategy `PROXY-PROT`, then the only directive to use would be:

```
Protect * PROXY-PROT
```

3.10.3 Using the htadm Command to Manage Password Files

The `htadm` command allows you to manage the caching and filtering proxy server password files. This command is also used to manage password files in Lotus Domino Go Webserver. Using the `htadm` command, you can add or delete a user ID and password, check a user's password, and create an empty password file. Passwords are stored encrypted.

Notice that, on AIX, the `htadm` executable file is automatically installed in the `/usr/sbin` directory, which is included in the `PATH` system environment variable. On Windows NT, an executable file, named `HTAdm.exe`, is stored in the `Bin` directory under the caching and filtering proxy server directory `WWW`. The `Bin` directory contains other executable files, and is automatically included in the `Path` system environment variable during the installation of the Caching and Filtering component.

In our case, we first created an empty password file, named `proxyUsers.pwd`, in the directory `/usr/lpp/internet/server_root/protect` of the caching and filtering proxy server installed on our AIX machine `venus`. To do this, we entered the following two commands:

```
cd /usr/lpp/internet/server_root/protect
htadm -create proxyUsers.pwd
```

Then we added to the `proxyUsers.pwd` password file a user ID `enzo` with password `enzo` and a string `User 1` as a comment. We did this by issuing the command:

```
htadm -adduser proxyUsers.pwd enzo enzo "User 1"
```

In a similar way, we added also a user ID `tatiana` with password `tatiana` and a string `User 2` as a comment. This was the command we entered:

```
htadm -adduser proxyUsers.pwd tatiana tatiana "User 2"
```

Notice that password file management can be accomplished also through the Configuration and Administration Forms.

The experience we have described in this section was performed on AIX, but it would be very similar on Windows NT.

3.10.4 Testing the Configuration

After configuring the proxy access protection, we wanted to verify that our configuration worked correctly.

On a Web client AIX machine, with IP address `172.16.1.5` and hostname `aixncf157`, we configured a Web browser to use our caching and filtering proxy server `venus` as proxy. The Web browser installed on `aixncf157` was Netscape Navigator Version 4.04 for AIX.

Then from the Web browser we requested the home page from the Web server `rs600030`. This we did by entering `http://rs600030` in the Location field of the browser. After this, we immediately got a prompt from the proxy server `venus` and we were requested to authenticate with a valid user ID and password. In fact, according to our configuration, we could not access the requested URL through the defined proxy server, unless we authenticated correctly with basic authentication.

We first entered user ID `otheruser` and password `otheruser` that had not been defined in the `proxyUsers.pwd` file, as shown in the following panel:



Figure 140. Entering a Wrong User ID and Password

Note that the prompt panel reminds you that the protection setup is named `ProxyServer`, according to the value we had defined for the `ServerID` subdirective.

Then we clicked **OK**, and got the following error authentication question displayed by our Netscape Navigator browser:

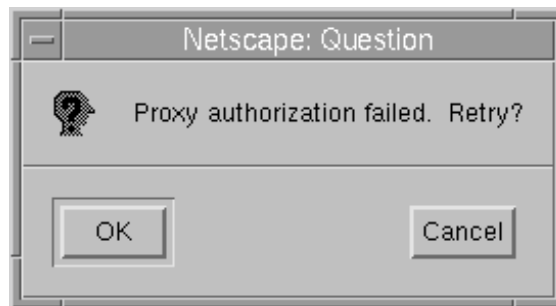


Figure 141. Proxy Authentication Failed

We clicked **OK** to retry, and then we tried to authenticate with a valid user ID and password. We entered user ID `tatiana` and password `tatiana`, since we had defined such a user ID and password in the `proxyServer.pwd` file. The following figure shows the Password prompt:

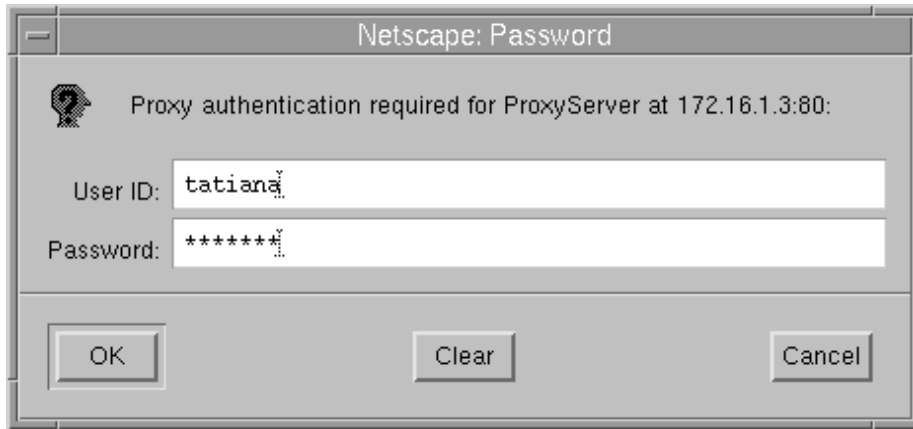


Figure 142. Entering a Valid User ID and Password

We clicked **OK**, and this time our authentication to the caching and filtering proxy server venus was successful. The proxy server gave us the authorization to use the proxy function and we could connect to the remote Web server rs600030, that sent back to us the Lotus Domino Go Webserver front page, as shown in the following screen:

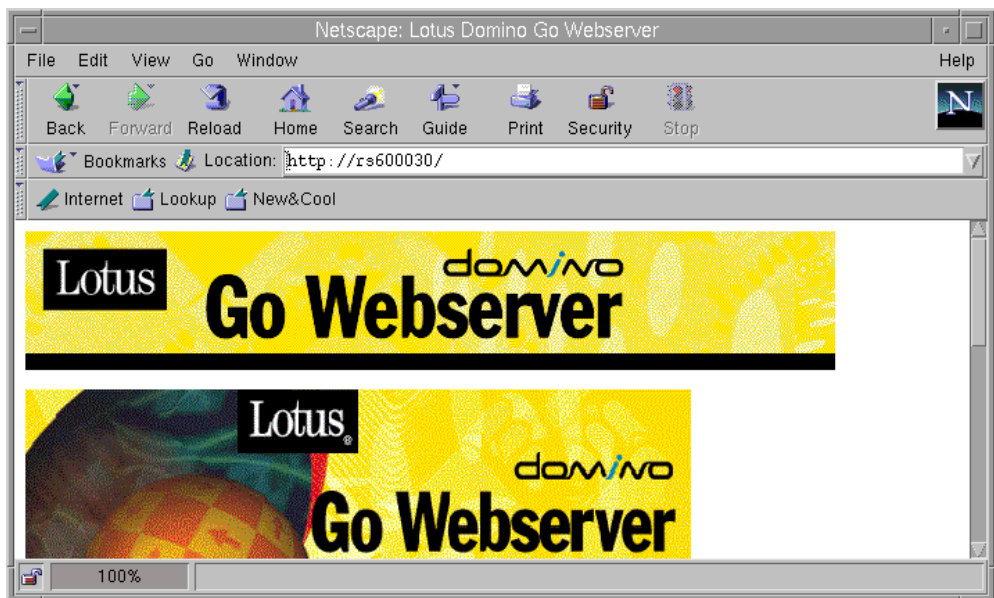


Figure 143. Confirmation of the Successful Authentication

This confirmed that we had performed a successful configuration of the proxy protection.

3.11 Platform for Internet Content Selection Protocol

In 3.4, "PICS-Based Filtering" on page 116, while we were describing the main features of the Caching and Filtering component of IBM WebSphere Performance Pack, we mentioned that this component allows you to apply PICS-compliant Web content filtering according at the proxy level. In this sense, we could also

say that the Caching and Filtering component of IBM WebSphere Performance Pack is PICS-enabled.

In 3.12, “PICS Scenario” on page 186, we show you how to implement a working PICS environment. However, before going on, it would be a good idea to explain more details about the PICS protocol, its terminology and the architecture it involves, so that the description of our experience will be clear.

3.11.1 What Is the PICS Protocol?

Platform for Internet Content Selection (PICS) is a technological standard that builds up a content labeling and filtering system for Web information. It is formed by a set of specifications used to create and manage ratings for the information published on any given Web site. The idea behind PICS is relatively simple. Since people have all kinds of different preferences and values, there should be a ratings and labeling standard that people can use to choose what content they would welcome seeing, or wish to exclude, based on certain parameters.

PICS development began in mid-1995, when the computing and online industries became sensitized to the possibility of online content censorship by the U.S. government and other governments around the world. Working under the aegis of the World Wide Web Consortium (W3C), the interested parties came together to create a technological solution that would support different rating systems. The PICS standard was adopted in May 1996. For the most up-to-date PICS information, see the World Wide Web Consortium's PICS Web site at the URL <http://www.w3.org/PICS/>.

3.11.2 PICS Content Labels

To have a common content rating and filtering system (such as the one described by PICS) effectively working, it is necessary that the main actors in the Web world (Web clients on one side, and Web servers on the other) cooperate in a synergical way. Web clients are required to use PICS-compliant browsers or PICS-compliant browser-side software to check the content of a Web site, and Web sites administrators are required to label their documents using the PICS standard.

Documents should be labeled according to the PICS specifications for the labels, as they are published at <http://www.w3.org/TR/REC-PICS-labels> by the World Wide Web Consortium. A PICS *content label* associated to a document is a piece of information that describes rating information about that document. It contains the rating of the document in reference to various dimensions or categories and specifies who has provided the label and when, along with other information.

3.11.3 Rating Systems

The PICS protocol defines a standard about how the ratings should be specified and transmitted, but it does not standardize the criteria by which you can rate the documents content. What PICS means to standardize is the method to rate the information. Such a method is also known as *rating system*. A rating system specifies the dimensions used for labeling, the scale of allowable values on each dimension, and a description of the criteria used in assigning values.

For example, a certain rating system can rate the humoristic content in a document based on a single dimension or category named Humor, with allowable values 0, 1, 2, 3, 4 and 5.

3.11.4 Rating Services

PICS does not intend to define an objective system of preferences. In other words, whoever can establish its own criteria to rate the information on the Web.

If you are a webmaster, you can decide on a rating system, rate your documents by creating the rating labels and then publish the documents to the world. What you should do is make your documents compliant to the PICS standard, so that your ratings can be understood by PICS-compliant programs on any platform, anywhere in the world.

Ratings can be assigned to a specific document, a group of documents or even to an entire Web site. The only problem now is that Web clients need to be sure that the ratings assigned correspond to what they expect. How can they trust you? How can they make sure that your rating criteria, meaning, your own rating system, meet their own preferences? We give an answer to these questions in the next subsections.

3.11.4.1 How Labels Are Provided

Typically, Web sites do not rate themselves although they could, considering that the PICS specifications do not forbid them to do that. What Web sites usually do is ask to be rated by a third party, called a *rating service*. A rating service is an entity that evaluates Web content according to their own published, well-known criteria. As a webmaster, you can contact one of the rating services to request assistance in assessing and labeling your own site and documents. The World Wide Web Consortium publishes a list of PICS self-rating services at the URL <http://www.w3.org/pub/WWW/PICS/selfrat.htm>.

Typically, after choosing a self-labeling service, you can connect to its Web site and describe the content of a single document, a group of documents or your entire Web site you want them to rate. You do this by filling out an online form. After submitting the form, the service provides you with a text label. At this point there are three possibilities:

1. The text label provided by the rating service can be dynamically embedded in the HTTP headers of the documents to be rated when clients request those documents.

The Web server administrator should be aware of whether their Web server software supports transmitting PICS labels in HTTP headers. If such a support is confirmed, the text information can be used to create PICS-compliant rating labels embedded in the HTTP header. The Web server administrator should store the labels in the Web server file system and use the PICS configuration file to manage and transmit them.

This is the preferred method for transmitting PICS labels, since it can be done automatically and does not add a significant overhead. The labels are sent along with the Web pages when a client requests them.

Notice that Lotus Domino Go Webserver supports this technology. It allows you to store the rating labels for all the documents on your Web site and

manage them from a central file. In this way rating labels can be embedded dynamically in the HTTP headers of the requested documents.

Another advantage of the technology we have described here is that security mechanisms such as message digests and digital signatures can be incorporated in the label creation to grant label validity.

2. The text label provided by the rating service can be statically embedded in the HTML headers of the documents to be rated. This operation must be accomplished when documents are created, or before clients requests them.

If the Web server software does not yet support the HTTP extensions for PICS, the Web server administrator can use `<META>` tags in the HEAD section of all the HTML files to store the PICS labels. This means that the Web server administrator should edit each of the HTML files that need to be rated and insert the rating information in the HEAD sections.

This process is entirely manual and therefore time-consuming, error-prone and difficult to maintain. It does not allow you to incorporate any of the security mechanisms that could guarantee the validity of the labels, such as message digests and digital signatures.

3. Independent rating services may make use of some procedures or software tools to examine particular URLs and create labels describing those URLs. The rating service stores the labels and distributes them from a separate server, called the *label bureau*. In this case a rating server also operates a label bureau server. Filtering software residing at the client-side needs to be instructed to check at that label bureau to find the labels.

A label bureau server provides labels dynamically when a client sends a specific request. In other words, if the user's profile asks for labels from a label bureau server, those labels will be requested in parallel with the content.

Notice that some clients might accept rating information that is imbedded in the file, but others might require a separate label from a registered rating service and a guarantee that it was created by that service. In other cases, Web clients might decide to contact the rating service only if the label information is not embedded with the requested document from the server. Then it might send a subsequent request directly to the rating service asking for the label information for that document.

Although this technology allows you to incorporate security mechanisms such as message digests and digital signatures to guarantee the validity of the labels, it requires a second connection, which takes longer and can discourage future visits to a specific Web site. The browser needs to wait until the label information is returned before it displays any data. Faster response data is the main reason why rating labels for a site should reside at the site itself.

No restrictions are applied in any of the three processes described above, in that anyone can label any document from any site. The PICS specification does not determine who can or will act as a rating service.

Many new rating systems were created in the last period by organizations with no interest in providing filtering software. Web users can elect to trust one or more of those organizations, and can decide to accept only ratings supplied by the organizations they trust.

A rating service can choose any criteria on which to rate Web sites. While some might rate Web sites for their violence or sexual content, others could choose to rate on educational content, political correctness, or even how *cool* the site is. Also, a rating service can rate any and all Web sites it wants to rate.

The two leading rating systems, Recreational Software Advisory Council on the Internet (RSACi) and SafeSurf, are based on PICS. The RSACi implements PICS labels using a number rating scale. The RSACi is an independent, non-profit organization based in Washington, D.C., that empowers the public, especially parents, to make informed decisions about electronic media by means of an open, objective, content advisory system. IBM is a corporate sponsor of RSACi and is represented on the RSACi Board of Directors.

So RSACi acts as a rating service which operates also as a label bureau. The RSACi rating system rates parameters such as nudity, graphic language, sexual content, and violence on a 0 (lowest) to 4 (highest) scale. For example, if you want to block sites with violence, but think some foul language is acceptable, you can have a filter accept sites with a violence maximum of 2, and a language maximum of 3.

3.11.4.2 URL Identifiers

Each rating system is identified by a valid URL. This enables several services to use the same rating system and refer to it by its identifier. The URL naming a rating system can be accessed to obtain a human-readable description of the rating system, such as the categories, scales, and intended criteria for assigning ratings. The URL that identifies a rating system is also used to advertise the rating services that use that particular rating system.

A rating service is identified by a URL as well. This identifier is included in all the labels assigned by the rating service. Since the service identifier is a URL, it can be used to retrieve a document. That document can be in any format, but PICS specifications recommend that it be in HTML format and give a description not only of the rating service, but also of the rating system. That document should at least provide a link to another document describing the rating system.

The rating service assigns labels according to some rating systems, and then distributes them, mainly via a label bureau. A label bureau is also identified by a URL, which the PICS-compliant client software should be able to contact to request the labels. In this sense, you can think of a label bureau as a computer system that supplies, via a computer network, ratings of documents.

3.11.5 PICS Filtering at the Proxy Server Level

PICS is a standard that lists rules for rating the information contained on any given Web site. Rating decisions are made on the base of particular categories, usually violence and pornography. Some Web browsers are PICS-compliant and are able to filter content information as content is received from the destination server. However, relying on browser settings may not be a safe solution, because such settings can be adjusted at the browser, and may be easily compromised.

PICS-Compliant Web Browsers

Microsoft Internet Explorer Version 4.0 supports the PICS protocol and can effectively block access to specific sites, allowing only specific sites, depending on software configuration. Netscape Navigator has not been made PICS-compliant yet.

With the Caching and Filtering component of IBM WebSphere Performance Pack, you can implement PICS filtering at the proxy server level. This removes the responsibility away from the client and your proxy server administrators can directly prevent certain types of information from being served to specific browsers (or to groups of browsers). Browsers with PICS filter settings defined will be then able to perform further filtering of the Web pages they will be served. This method ensures that clients will only get the level of content specified at the proxy. Interaction between the client and the proxy administrator would be required to change the sensitivity of the filter.

The diagram below shows the logical flow of the PICS filtering at a proxy server level:

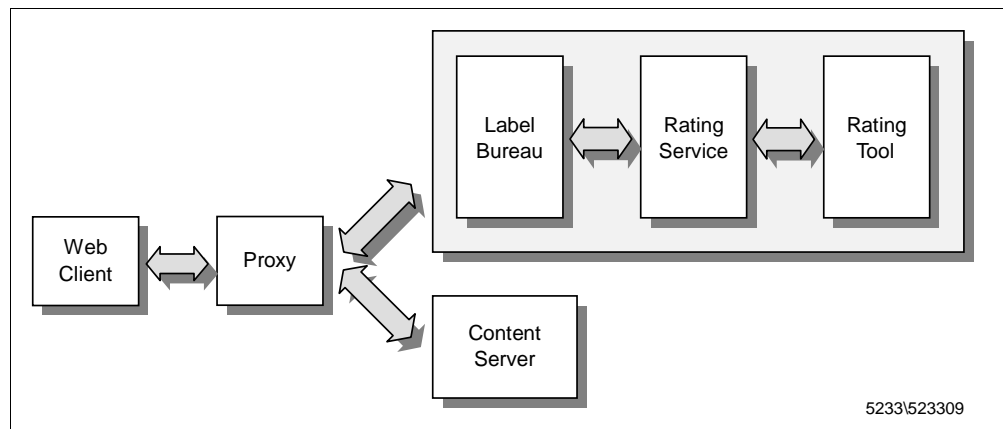


Figure 144. PICS Filtering at the Proxy Server Level

The proxy server administrator has set up and enabled a filtering profile on the proxy server. To decide whether a particular document will be passed or blocked, the proxy filtering profile uses the values contained in the PICS labels supplied by a label bureau server, which is in turn managed by a rating service. The rating service might own a rating tool to examine particular URLs and create labels describing those URLs. This rating tool can implement a procedure to discover a new site as soon as it goes online, examine the site and store a PICS label at the label bureau server.

Later, when content from this site is requested by the user's client software, the PICS label is requested from the label bureau server by the proxy. If the profile establishes that the values in this PICS label mean that the content is not wanted, then the proxy will send an HTML page back to the client, explaining the reason why the content is not being delivered. It may or may not include instructions on how to bypass the blocking or how to initiate the correction of a faulty rating if necessary. However, in most cases, the values in the label mean the content is

acceptable, and the content is fetched from the content server to the proxy and then forwarded to the user.

If the proxy caching is enabled, during further fetches the user sees no delay, since the proxy server caches only pages that have previously been accepted, and then serves them directly from its cache.

The label bureau server might tell the proxy that this site has not yet been rated, and the profile determines whether the user is sent the content anyway or a `Not Yet Rated` page by the proxy is sent. The reviewer's tool is notified by the label bureau that this site should be rated as soon as possible. Some label bureau servers in the future will fetch the content and run a program to create an interim rating that will be returned, pending the reviewer's more accurate site evaluation.

The diagram shown in Figure 144 on page 185, which we have just explained, is simplified if the Web server is enabled to embed PICS labels in the Web documents it serves (either in the HTTP header or in the HTML header), and the proxy server is configured to accept such labels without the need to contact an external label bureau. In the next section we show you how to implement a working PICS environment, and discuss both the possibilities where the PICS labels are provided by an external label bureau or by the Web server itself.

3.12 PICS Scenario

In this section we describe how to build a PICS environment where the content filtering is performed at the proxy server level using the Caching and Filtering component of IBM WebSphere Performance Pack.

We show you a working scenario where the PICS labels generated by a rating service are provided by an associated label bureau (see 1 on page 182). Then we will show you another example where the PICS labels are embedded in the Web documents at the Web server level (see 2 on page 183 and 3 on page 183).

These experiences were performed using the following components:

1. Netscape Navigator 4.06 was installed on a Windows NT Server 4.0 client machine.
2. The Caching and Filtering component of IBM WebSphere Performance Pack 1.0, installed on a Windows NT Server 4.0 machine, had the role of the caching and filtering proxy server.
3. Lotus Domino Go Webserver 4.6.2.5 installed on an AIX 4.3 machine had the role of the Web server.
4. Lotus Domino Go Webserver 4.6.2.5 installed on an AIX 4.3.1 machine had the role of the rating service and label bureau of our architecture.

The purpose of this section is to demonstrate the capability of the Caching and Filtering component of filtering Web content. At the end of the section, it will be clear that content filtering, when applied at the proxy server level rather than at the Web client level, guarantees security and reliability.

3.12.1 Network Configuration

The following table summarizes the features of the environment we used:

Table 3. PICS Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	Web Client
IBM PC 365	computer2	9.24.104..212	Windows NT Server 4.0	Caching and Filtering Proxy Server
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3.1	Web Server
IBM RS/6000 43P	august	9.24.104.46	AIX 4.3.1	Rating Service and Label Bureau

The architecture of the PICS environment we built is shown in the following diagram, which you can easily compare with the general architecture diagram shown in Figure 144 on page 185:

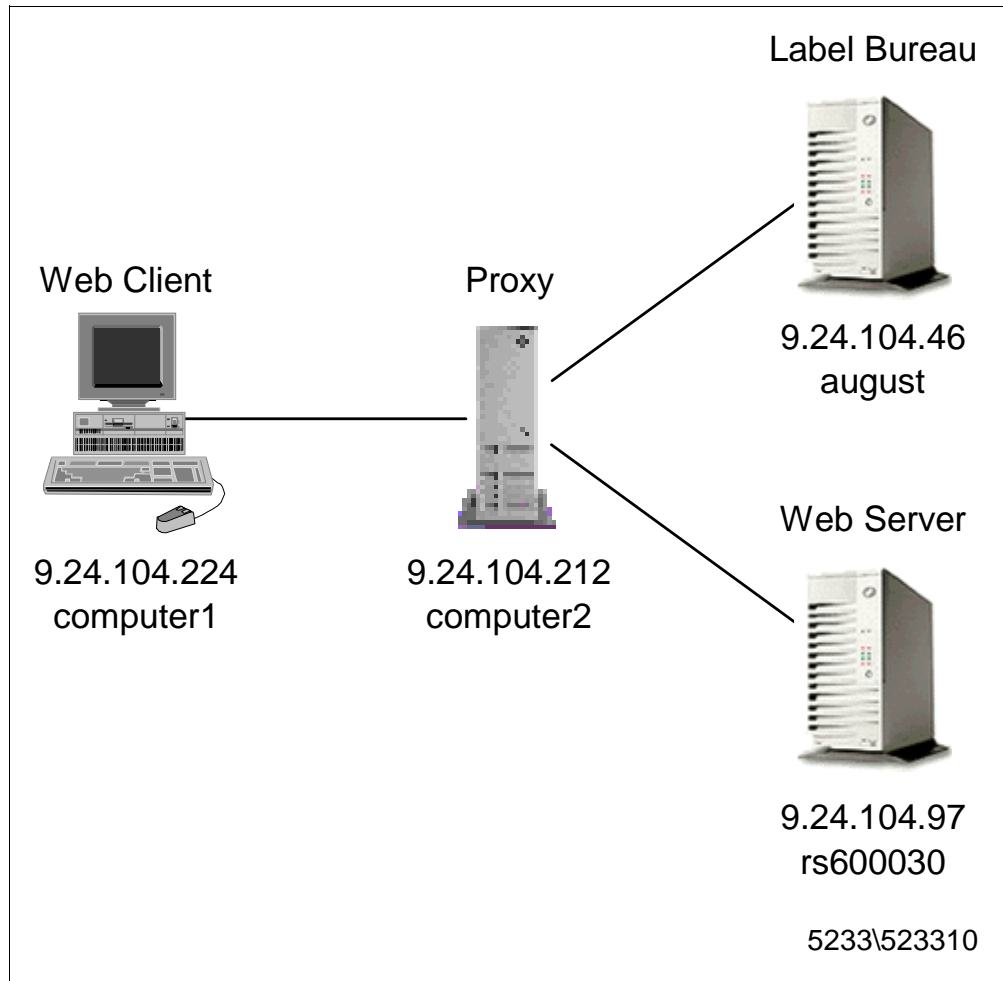


Figure 145. PICS Scenario Environment

The flow in a PICS environment where content filtering is applied at the proxy server level is described in 3.11.5, "PICS Filtering at the Proxy Server Level" on page 184.

To show how PICS works, we built up an architecture including all the parts that are interested in a PICS flow. In the rest of this section, we give you more details on the machines we used.

The Web client machine was an IBM PC 365, with a CPU of 166 MHz and 64MB of RAM. As we have already said, its operating system was Windows NT Server 4.0, but we had also applied Service Pack 3. The Web browser functionality was supplied by Netscape Navigator 4.0.6.

The caching and filtering proxy server machine was an IBM PC 365, with a CPU of 166 MHz and 64MB of RAM. The operating system running on this machine was Windows NT Server 4.0, where we had also applied Service Pack 3. The caching and filtering proxy server functionality was supplied by the Caching and Filtering component of IBM WebSphere Performance Pack Version 1.0.

The Web server machine was an IBM RS/6000 43P, with a CPU of 333 MHz and 196MB of RAM. The operating system running on this machine was AIX 4.3. The content server functionality was supplied by Lotus Domino Go Webserver Version 4.6.2.5.

The rating service and label bureau machine was an IBM RS/6000 43P, with 62.5 MHz of CPU and 64MB of RAM. The operating system running on this machine was AIX 4.3.1. The rating service and label bureau functionality was supplied by Lotus Domino Go Webserver Version 4.6.2.5.

All the above machines were connected to the same token-ring network.

3.12.2 Configuration of the Web Server

First of all, we set up our own Web site, managed by Lotus Domino Go WebServer Version 4.6.2.5. On the Web site we decided to publish, among others, the following four HTML documents at the URL <http://rs600030.itso.ral.ibm.com/PICSxmp>:

1. age1.html
2. age2.html
3. age3.html
4. age4.html

These files are a subset of the sample files distributed with Lotus Domino Go Webserver. After a default installation on AIX, they are located in the directory `/usr/lpp/internet/server_root/pub/PICSxmp`.

Acting as webmasters of this site, we chose to have the mentioned HTML pages be rated by a rating service. As we said in 3.11.4, "Rating Services" on page 182, a rating service evaluates Web content according to its own published criteria and then can distribute the labels through a label bureau.

3.12.3 Configuration of the Rating Service

First, we had to choose the rating service and the label bureau.

Lotus Domino Go Webserver can be configured to act not only as a regular Web server, but even as a rating service and label bureau, since it can store rating labels even for other Web sites and serve them in response to client requests.

So we used Lotus Domino Go Webserver 4.6.2.5 on the machine with host name august and created our rating service and label bureau on that machine.

To set up our rating service and label bureau machine, we first had to define our own rating system, or, in other words, to specify the rating criteria. To do that, we needed a PICS-compliant rating system description file that described the rating system used to rate the documents. These kinds of files are called RAT files. A rating service must provide a RAT file along with the rating labels for the HTML documents.

To see how you can create your own RAT file, which should carry a .rat extension, check the World Wide Web Consortium's PICS specifications for rating services and rating systems at the URL <http://www.w3.org/pub/WWW/PICS/services.html>. It includes the syntax for the machine-readable format of RAT files.

The Lotus Domino Go Webserver fileset includes a sample RAT file, called coolness.rat and located in the directory /usr/lpp/internet/server_root/labels. We created the directory /usr/lpp/internet/server_root/enzoRatings and copied the coolness.rat file into this new directory. Then we made some light changes to the coolness.rat file, in that we essentially changed the second, third and fourth line to personalize that file according to our platform. The entire file is shown in the following two figures:

```

((PICS-version 1.1)
(rating-system "http://august.itso.ral.ibm.com/enzoRatings/EnzoRS.html")
(rating-service "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html")
(name "The Enzo Rating System")
(description "This rating system is based on sample provided with Domino Go
WebServer, It categorizes three important criteria: coolness, age-range,
and number of graphics.")
(category
(transmit-as "Coolness")
(name "Coolness Index")
(label
(name "Way Cool")
(description "This site is majorly cool")
(value 1))
(label
(name "cool")
(description "Pretty cool")
(value 2))
(label
(name "Mediocre coolness")
(description "Tries to be cool but falls a little short")
(value 3))
(label
(name "Not cool")
(description "Like the name, not cool")
(value 4))
(label
(name "Totally Uncool")
(description "This site is a waste of time")
(value 5)))
(category
(transmit-as "Age-range")
(name "age-range")
(label
(name "All ages")
(description "This site is suitable for everyone")
(value 1))
(label
(name "Teenager and older")
(description "Teenagers and older")
(value 2))
(label
(name "Adult only")
(description "For adults only")
(value 3))
(label
(name "No one")
(description "No one should see this site")
(value 4)))
(category
(transmit-as "Graphics")
(name "number of graphics")
(label

```

Figure 146. (Part 1 of 2). coolness.rat File

```

(name "0 to 1")
(description "Hardly any graphics")
(value 1))
(label
(name "1 to 5")
(description "A few graphics")
(value 2))
(label
(name "10 to 20")
(description "A lot of graphics")
(value 3))
(label
(name "20+")
(description "Plan to be here all day")

```

Figure 147. (Part 2 of 2). coolness.rat File

Now, we explain to you what the meaning of the above sample RAT file.

1. The URL identifier of the rating system used is <http://august.itso.ral.ibm.com/enzoRatings/EnzoRS.html>. The Web page pointed by the above URL is shown in the following figure. The document available at that URL should be a human-readable description of the categories, scales, and intended criteria for assigning ratings, as we also explained in 3.11.4.2, "URL Identifiers" on page 184:

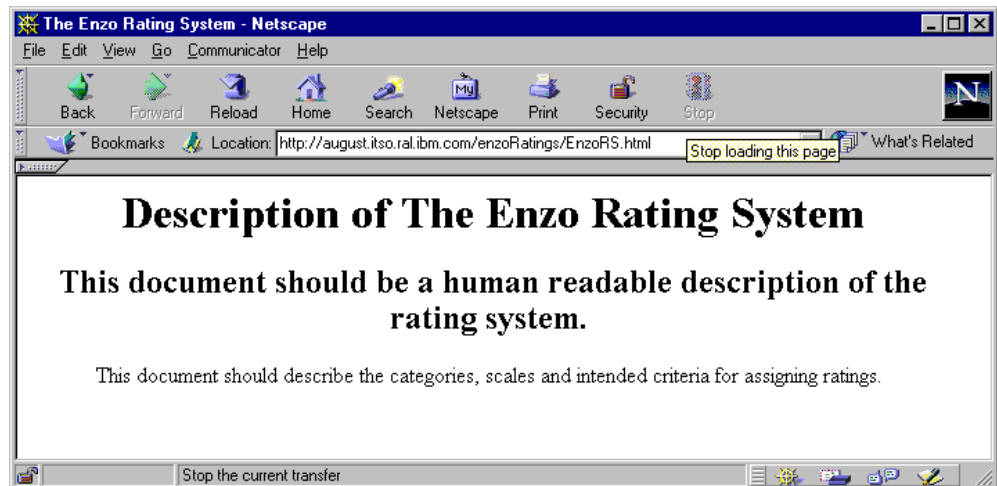


Figure 148. Web Page Pointed by the Rating System URL Identifier

2. The URL identifier of the rating service is <http://august.itso.ral.ibm.com/enzoRatings/V1-0.html>. The Web page pointed by the above URL is shown in the following figure. The labels themselves will have this URL in them to identify the rating service that created them. The document available at this URL should be a human-readable description of the rating service, as explained in 3.11.4.2, "URL Identifiers" on page 184:

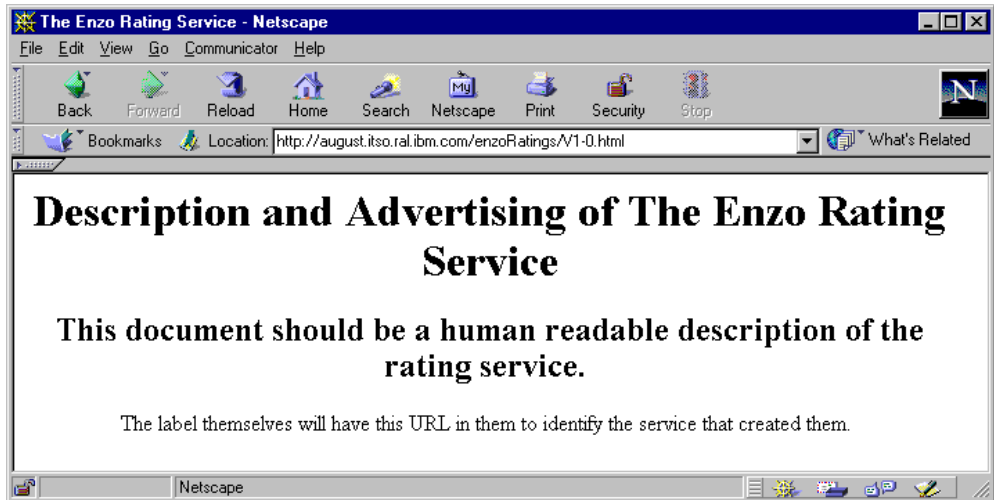


Figure 149. Web Page Pointed by the Rating Service URL Identifier

URL Identifiers Accessibility

In order to make the URLs specified for the rating-service and rating-system parameter accessible from Web browsers, we inserted the following Pass directive in the httpd.conf configuration file of our Web server AIX machine rs600030:

```
Pass          /enzoRatings/*    /usr/lpp/internet/server_root/enzoRatings/*
```

3. The Enzo Rating System is the name of the rating service.
4. There are three top-level categories in this rating system. Each category has two names: a short transmission name to be used in labels, and a name that is more easily understood and usually longer than the transmission name.

For example, the first one has transmission name `Coolness` and the longer name `Coolness Index`. The second one has a transmission name `Age-range` and name `age-range`. The third one has transmission name `Graphics` and name `number of graphics`.

5. All the categories only allow the integer values 1, 2, 3 and 4. Each label attribute provides a label with a value and associates a name and a description to the value. In a content filtering operation you can use the value or the name of a label, but it is recommended to use the value rather than the name of a label, in order to be independent from the language and to make it easier to write rules expression filtering.

Once we had modified and personalized the RAT file describing the rating system we intended to use, we needed to create the labels for the documents to be rated, and then store the labels in the label bureau server's file system. Actually, since in this case we were acting as rating service administrators, we should have used some rating tools to rate the documents, as explained in 3.11.5, "PICS Filtering at the Proxy Server Level" on page 184.

Let's assume that we used our own rating tool to produce the following label files for the documents `age1.html`, `age2.html`, `age3.html` and `age4.html` respectively:

1. age1.lbl
2. age2.lbl
3. age3.lbl
4. age4.lbl

We do not need to use any rating tools because Lotus Domino Go WebServer already provide such files, which on AIX come in the directory /usr/lpp/internet/server_root/labels after a default installation of the product.

We copied those label files into the directory where we had already copied the RAT file, which was /usr/lpp/internet/server_root/enzoRatings. Then we made some little changes to the label files, in that we essentially changed the first line of each file to personalize the label according to our system configuration. The label files age1.lbl, age2.lbl, age3.lbl and age4.lbl are shown in the following four figures respectively:

```
(PICS-1.1 "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
l for %%URL%%
comment "This site is suitable for everyone"
on "1996.04.04T08:15-0500"
r (Coolness 0 Age-range 1 Graphics 0))
```

Figure 150. age1.lbl File

```
(PICS-1.1 "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
l for %%URL%%
comment "For teenagers and older"
on"1996.04.04T08:15-0500"
r (Coolness 0 Age-range 2 Graphics 0))
```

Figure 151. age2.lbl File

```
(PICS-1.1 "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
l for %%URL%%
comment "For adults only"
on "1996.04.04T08:15-0500"
r (Coolness 0 Age-range 3 Graphics 0))
```

Figure 152. age3.lbl File

```
(PICS-1.1 "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
l for %%URL%%
comment "No one should see this site"
on "1996.04.04T08:15-0500"
r (Coolness 0 Age-range 4 Graphics 0))
```

Figure 153. age4.lbl File

The above label files have a very similar common structure.

1. They specify through the string PICS-1.1 that the label structure complies to the PICS label specifications Version 1.1, contained in the document PICS Label Distribution Label Syntax and Communication Protocols Version 1.1, found at the URL <http://www.w3.org/TR/REC-PICS-labels>.
2. They report the URL identifier <http://www.w3.org/TR/REC-PICS-labels> of the rating service that has created the label. We remind you that this URL is the same URL specified as a value of the `rating-service` parameter in the RAT file.
3. They show which document the label was created for. In this case, the short version `l` for the keyword `labels` has been used. The value after the keyword `for` should be the URL of the document that the label refers to. So, for example, in the first label file `age1.lbl`, which refers to the `age1.html` document, the full line should be:

```
labels for "http://rs600030.itso.ral.ibm.com/PICSxmp/age1.html
```

Lotus Domino Go Webserver has added extensions to the format of the labels specified at <http://www.w3.org/TR/REC-PICS-labels>. One of these extensions allows you to insert in label files some variables, such as `%%URL%%`, the current URL will be substituted for this variable. This means that when the label bureau server receives a request for a rating label that contains the variable `%%URL%%`, it replaces this variable with the correct current URL before sending the label.

4. They may specify some comments for the label by using lines starting with the `comment` statement. These comment lines are sent back to the clients. Another extension of Lotus Domino Go Webserver allows you to insert another type of comment for your own use into the label files, by beginning the comment lines with the pound character `#`. Lines beginning with the `#` character are not sent back to the clients, but are for the use of the rating service and label bureau administrator. This type of comment is an addition to the `comment` statement used inside labels.
5. They report the date on which this rating label was issued.
6. Finally, they specify the value of the rating assigned to the document for each category specified in the RAT file. Notice that it is possible to use the short version `r` for the key word `ratings`.

If we decided to use the complete key words and replace the `%%URL%%` variable with its actual value, a label file (for example, `age1.lbl`) should appear similar to the following:

```
(PICS-1.1 "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
labels for "http://rs600030.itso.ral.ibm.com/PICSxmp/age1.html
comment "This site is suitable for everyone"
on "1996.04.04T08:15-0500"
ratings (Coolness 0 Age-range 1 Graphics 0))
```

Figure 154. `age1.lbl` File - Complete Version

3.12.4 Configuration of the Label Bureau

At this point, by creating the RAT file `coolness.rat`, we established the rating system we intended to apply, and we also published that rating system through the URL <http://august.itso.ral.ibm.com/enzoRatings/EnzoRS.html>. We also

published the rating service at the URL <http://august.itso.ral.ibm.com/enzoRatings/V1-0.html> and finally we rated some documents of the Web site <http://rs600030.itso.ral.ibm.com>.

Now we show you the steps to configure Lotus Domino Go WebServer as a PICS label bureau in order to serve the stored rating labels in response to client requests.

As the first step, we had to inform the server that it would act as a PICS label bureau, and we had to specify where to direct the PICS rating label requests. To do so, we added the following `Service` directive to the configuration file `httpd.conf` of our rating service and label bureau AIX machine:

```
Service /Ratings INTERNAL:PICS-Ratings
```

We entered `/Ratings`, so clients who wanted to request labels from our label bureau should have requested the URL <http://august.itso.ral.ibm.com/Ratings>. In your configuration, you will replace `/Ratings` with the relative path of the URL you will use on the label bureau server for label requests. For example, if for label requests you publish the URL <http://www.coolratings.com/CoolSite>, you would only include `/CoolSite` in the `Service` directive.

As the second step, we had to tell our server which documents had been rated, what host would serve them, and where the labels could be found in the label bureau server machine file system. We specified that by editing the PICS configuration file `/etc/ics_pics.conf`, which is used to associate the rated documents to the specific label files. The `/etc/ics_pics` file we used on our system is shown in the following figure:

```

#
# COMPONENT_NAME: web ics_pics.conf
#
# FUNCTIONS:
#
# ORIGINS: 10 26 27
#
# (C) COPYRIGHT Lotus Development Corporation 1997
# (C) COPYRIGHT International Business Machines Corporation 1997
# All Rights Reserved
#
# This software is subject to the Lotus Software Agreement
# Restricted Rights for U.S. government users, and applicable export
# regulations. Lotus is a registered trademark and Lotus Domino Go Webserver
# is a trademark of Lotus Development Corporation.

#
#       Sample PICS Configuration file for
#       Lotus Domino Go Webserver
#
# Please use the remote administration interface to add labels
# to your documents.

DefineLBSERVICE "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html" "The Enzo Rating System"
/usr/lpp/internet/server_root/enzoRatings/coolness.rat {
LABELFILE /usr/lpp/internet/server_root/enzoRatings/age1.lbl This site is suitable for everyone
LABELFILE /usr/lpp/internet/server_root/enzoRatings/age2.lbl Teenagers and older
LABELFILE /usr/lpp/internet/server_root/enzoRatings/age3.lbl For adults only
LABELFILE /usr/lpp/internet/server_root/enzoRatings/age4.lbl No one should see this site
LABELFILE /usr/lpp/internet/server_root/enzoRatings/cool1.lbl This site is majorly cool
LABELFILE /usr/lpp/internet/server_root/enzoRatings/cool2.lbl Pretty cool
LABELFILE /usr/lpp/internet/server_root/enzoRatings/cool3.lbl Tries to be cool but falls a little short
LABELFILE /usr/lpp/internet/server_root/enzoRatings/cool4.lbl Like the name, not cool
LABELFILE /usr/lpp/internet/server_root/enzoRatings/cool5.lbl Totally Uncool
LABELFILE /usr/lpp/internet/server_root/enzoRatings/graphics1.lbl Hardly any graphics
LABELFILE /usr/lpp/internet/server_root/enzoRatings/graphics2.lbl A few graphics
LABELFILE /usr/lpp/internet/server_root/enzoRatings/graphics3.lbl A lot of graphics
LABELFILE /usr/lpp/internet/server_root/enzoRatings/graphics4.lbl Plan to be here all day
}

LabelsFor http://rs600030.itso.ral.ibm.com http://august.itso.ral.ibm.com/enzoRatings/V1-0.html {
/PICSxmp/age1.html /usr/lpp/internet/server_root/enzoRatings/age1.lbl
/PICSxmp/age2.html /usr/lpp/internet/server_root/enzoRatings/age2.lbl
/PICSxmp/age3.html /usr/lpp/internet/server_root/enzoRatings/age3.lbl
/PICSxmp/age4.html /usr/lpp/internet/server_root/enzoRatings/age4.lbl
/PICSxmp/cool1.html /usr/lpp/internet/server_root/enzoRatings/cool1.lbl
/PICSxmp/cool2.html /usr/lpp/internet/server_root/enzoRatings/cool2.lbl
/PICSxmp/cool3.html /usr/lpp/internet/server_root/enzoRatings/cool3.lbl
/PICSxmp/cool4.html /usr/lpp/internet/server_root/enzoRatings/cool4.lbl
/PICSxmp/cool5.html /usr/lpp/internet/server_root/enzoRatings/cool5.lbl
/PICSxmp/graphics1.html /usr/lpp/internet/server_root/enzoRatings/graphics1.lbl
/PICSxmp/graphics2.html /usr/lpp/internet/server_root/enzoRatings/graphics2.lbl
/PICSxmp/graphics3.html /usr/lpp/internet/server_root/enzoRatings/graphics3.lbl
/PICSxmp/graphics4.html /usr/lpp/internet/server_root/enzoRatings/graphics4.lbl
}

```

Figure 155. ics_pics.conf File

The above ics_pics.conf configuration file contains two types of paragraphs:

1. The `DefineLBSERVICE` paragraph lists local label files associated with our own local bureau and rating service.

The first line of the paragraph consists of the keyword `DefineLBSERVICE`, the rating service URL, the quoted name of the rating service, the fully qualified name of the service's RAT file that describes the rating system, and an opening brace.

The body of the paragraph lists the fully qualified names of the label files associated with this service. Each file name is introduced by the keyword LABELFILE.

The paragraph ends with a closing brace.

The correct syntax would be:

```
DefineLBSERVICE servicename "name-of-service" ratingfile {
    LABELFILE /path/LabelFile1 "description"
    LABELFILE /path/LabelFile2 "description"
    ...
}
```

where:

- *servicename* is the name or URL identifier of the rating service.
 - *name-of-service* is a text string representing the name of the rating service.
 - *ratingfile* is the fully qualified name of the service's RAT file in the label bureau server machine file system.
 - */path/LabelFile1, /path/LabelFile2, ...* are the fully qualified names of the label files in the label bureau server machine file system.
 - *description* is a text description of the label.
2. The second paragraph in the `ics_pics.conf` file is `LabelsFor`. It specifies the ratings given by the rating service for documents on a given Web server.

The first line of the paragraph consists of the keyword `LabelsFor`, the name of the Web server on which the rated documents are found, the name of the rating service and an opening brace.

The body of the paragraph specifies labels for a set of documents. The paragraph ends with a closing brace.

The correct syntax would be:

```
LabelsFor servername servicename {
    /WebPath1/document1 /path/LabelFile1
    /WebPath2/document2 /path/LabelFile2
    ...
}
```

where:

- *servername* is a fully qualified URL of the remote servers on which the documents being rated are found. Note that the fully qualified URL must not end with a trailing slash; thus, `http://rs600030.itso.ral.ibm.com` is acceptable as a value of the *servername* variable hostname on a `LabelsFor` line, but `http://rs600030.itso.ral.ibm.com/` is not.
- *servicename* is the fully qualified URL to which clients will send their label requests.

- */WebPath1/document1, /WebPath2/document2, ...* are the Web paths and names of the documents being rated.

Notice that the path a Web client would use when requesting, for example, *document1* is given by adding */WebPath1/document1* at the end of the *servername* value, that is:

```
servername/WebPath1/document1
```

- */path/LabelFile1, /path/LabelFile2, ...* are the fully qualified names of the label files in the label bureau machine file system.

This completes the description of the label bureau configuration.

3.12.5 PICS Filter Configuration on the Proxy Server

Once we set up the label bureau we trusted to provide PICS labels, we had to set the PICS filters on the caching and filtering proxy server machine. We created our PICS filters by manually editing the *javelin.cnf* configuration file.

The following lines are extracted by the *javelin.cnf* configuration file. They are the lines we added to write our PICS filtering rules.

```
# ===== #
#
#     PICS Filtering directives
#
# ===== #

#     PICS Filtering using PICSRules
DefinePicsRule "FilterAge" {
    (PicsRule-1.0
    (
        passURL ("http://w3.ibm.com/*")
        failURL ("http://rs600023.itso.ral.ibm.com/*")
        passURL ("http://august.itso.ral.ibm.com/Ratings")
        serviceinfo (name "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
            shortname "TheEnzo"
            bureauURL "http://august.itso.ral.ibm.com/Ratings"
            ratfile "coolness.rat"
        )
        name (rulename "Filter based on Age"
            description "Fail the URL when you do not have the age"
        )
        ibm-javelin-extensions (
            active-days "1111111"
            start-time "01:00"
            end-time "23:00"
        )
        Filter ( Pass "(TheEnzo.Age-range < 3)" )
    )
    )
}
```

Figure 156. *javelin.cnf* File

In the *javelin.cnf* configuration file you specify the PICS filtering rules using the *DefinePicsRule* directive. It is possible to specify only one rule per *DefinePicsRule* directive.

Each rule should begin with the keyword *DefinePicsRule*, then the mnemonic name you want to assign to the rule and an opening brace. The body of the rule contains the definitions of the rule. Each rule must end with a closing brace.

All the statements that define the PICS rule are preceded by the keyword `PicsRule-1.0` and enclosed in opening and closing round brackets. Version 1.1 of the Caching and Filtering component, included in the IBM WebSphere Performance Pack 1.0, supports the Version 1.0 of the PICS Rules Specifications, and not the last version, 1.1. For this reason, we recommend that you use the syntax defined in Version 1.0 of the PICS Rules Specification, since the syntax specified in Version 1.1 would not be recognized.

The `passURL` and `failURL` statements allow you to specify all the URLs that the clients can or cannot view respectively. For example, we wanted to pass all the pages from the URL `http://w3.ibm.com`, and block all the pages coming from `http://rs600023.itso.ral.ibm.com`. To do this we inserted the following lines in the above section of the `javelin.cnf` configuration file:

```
passURL ("http://w3.ibm.com/*")
failURL ("http://rs600023.itso.ral.ibm.com/*")
```

To conditionally filter using PICS labels from a rating service, the rule must contain information about that service. First, a `passURL` statement is required to ensure that the label bureau URL used for label requests is admitted. This is the reason why, in our case, we inserted the line:

```
passURL ("http://august.itso.ral.ibm.com/Ratings")
```

The service information is preceded by the statement `serviceinfo` and is enclosed between opening and closing round brackets.

- The statement `name` allows you to specify the rating service URL identifier.
- The statement `shortname` allows you to specify a common name for the rating service. You will use this common name to refer to the rating service in the `Filter` statement, which is the last statement of the directive.
- The `bureauURL` statement allows you to specify the label bureau URL, which is the URL used on the label bureau server to serve label requests.
- The statement `ratfile` allows you to specify the RAT file.

A statement related to the filter name information follows. With `rulename` you indicate a string that is displayed to the user when the requested page is blocked. With `description` you can specify a text description field for the rule writer.

The next group of statements include extensions to the filtering PICS rules. Those extensions are not defined by the PICS Rules Specifications, but are specific to the Caching and Filtering component of IBM WebSphere Performance Pack. They can be used by the caching and filtering proxy server administrator to better define the filtering behavior. In our case, we instructed the caching and filtering proxy server to consider the filtering rule active all seven days of the week, each day from 01:00 to 23:00. Notice that hours are expressed in the format `hh:mm`. There were no particular reasons why we decided to limit the filter rule activity to that particular time frame, only testing purposes.

Finally, we have the `Filter` statement, by which:

- You can force the caching and filtering proxy server to read the values assigned in the PICS labels to one or more specific categories.
- You can define the logical operations on the rating values to be performed in order to decide if the URL should be passed or blocked.

In our case, the line:

```
Filter ( Pass "(TheEnzo.Age-range < 3)" )
```

- Forces the caching and filtering proxy server to read the value assigned in the PICS label to the category having the transmission name `Age-range`, which has been assigned to the document by the rating service having the parameter `shortname` set to `TheEnzo`.
- Instructs the caching and filtering proxy server to pass the URL if the value read is less than 3.

The same result would be obtained if the following line were used but there is a substantial difference between the two filters:

```
Filter ( Block "(TheEnzo.Age-range >= 3)" )
```

In fact both the filters will have the same results when a rating label on the `age-range` category is provided. The difference in their behavior is clear when an `age-range` rating for the page is unavailable.

In the `Pass` statement, the comparison must return `true` for the URL to be passed. Instead, in the `Block` statement, the comparison must return `true` for the URL to be blocked. Because the comparison returns `false` anyway when a Web page is not rated on the `age-range` category, the `Pass` statement would block the URL, while the `Block` statement would pass the URL.

Notice that order is very important in the configuration files. The process stops when it finds the first match, so if in the `javelin.cnf` configuration file shown in Figure 156 on page 198, you put

```
passURL ("http://w3.ibm.com/*")
```

after

```
passURL ("http://august.itso.ral.ibm.com/Ratings")
```

it would match the rating service first and check to see its ratings.

We want to mention that at the beginning of our experience, we tried to use the Configuration and Administration Forms instead of manually editing the `javelin.cnf` configuration file. We found the directions to do that in *Web Traffic Express for Multiplatforms User's Guide*, GC31-8645. We followed those directions and we defined the service information associated to our PICS service bureau. Then we opened the `javelin.cnf` configuration file and found that no `serviceinfo` statement had been inserted in the directive `DefinePicsRule`. Instead, we found a different directive at the end of the file:

```
DefineServiceInfo "TheBest" {
    serviceinfo (name "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
                shortname "TheEnzo"
                bureau "http://august.itso.ral.ibm.com/Ratings"
                ratfile "coolness.rat"
                )
}
```

When we tried to stop and restart the caching and filtering proxy server, we opened the error log file and here is what we found:

```
Restart succeeded
Server is ready
Unknown Javelin directive "defineserviceinfo"
```

An unexpected error message was displayed as soon as the proxy server restarted. We tried to make our tests anyway, but the filtering activity of the caching and filtering proxy server did not work.

However, this is a known problem that is present only in Version 1.1 of the Caching and Filtering component, included in IBM WebSphere Performance Pack Version 1.0. At the time this book went to print, Version 1.2 of the Caching and Filtering component became available, even if it wasn't included in IBM WebSphere Performance Pack 1.0, and we knew that this problem had been fixed.

So we could not make use of the Configuration and Administration Forms in this case, but the solution to manually edit the javelin.cnf configuration file, as we have described to you, was easy and successful anyway.

3.12.6 Working with PICS Filtering at the Proxy Server Level

Adopting the configuration we have shown so far, we performed some tests. Before proceeding, we stopped and restarted the Lotus Domino Go Webserver on the two machines rs600030 and august where it was running. We also stopped and restarted the caching and filtering proxy server that was running on the machine computer2. This operation we considered necessary to have all the configuration changes take effect.

Then, on the client machine, we configured the Web browser to access the Internet through the proxy server computer2.itso.ral.ibm.com. After that we requested the URL <http://rs600023.itso.ral.ibm.com/>. The machine rs600023.itso.ral.ibm.com was another Web server in our network environment. The filter should have not passed that page, since we had explicitly blocked it in the PICS filtering directive of the javelin.cnf configuration file. And in fact we got the following error message from the caching and filtering proxy server:

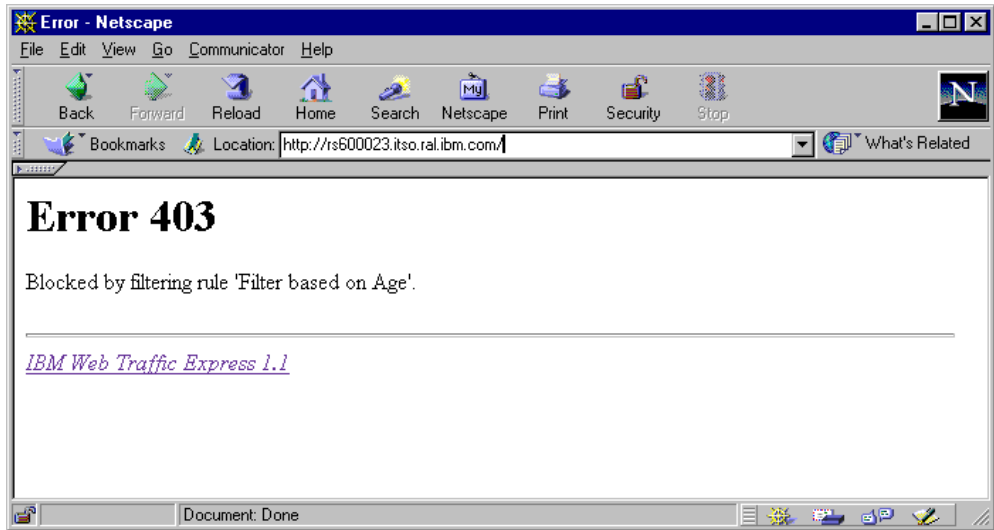


Figure 157. Demonstration of the failURL Directive

Note that the string `Filter based on Age` was the value of parameter `rulename` in the PICS rule (see Figure 156 on page 198).

We expected that all the pages from `rs600023.itso.ral.ibm.com` would be blocked by the PICS rule. In fact we requested the page `http://rs600023.itso.ral.ibm.com/PICSxmp/age1.html`, and the caching and filtering proxy server blocked this too:

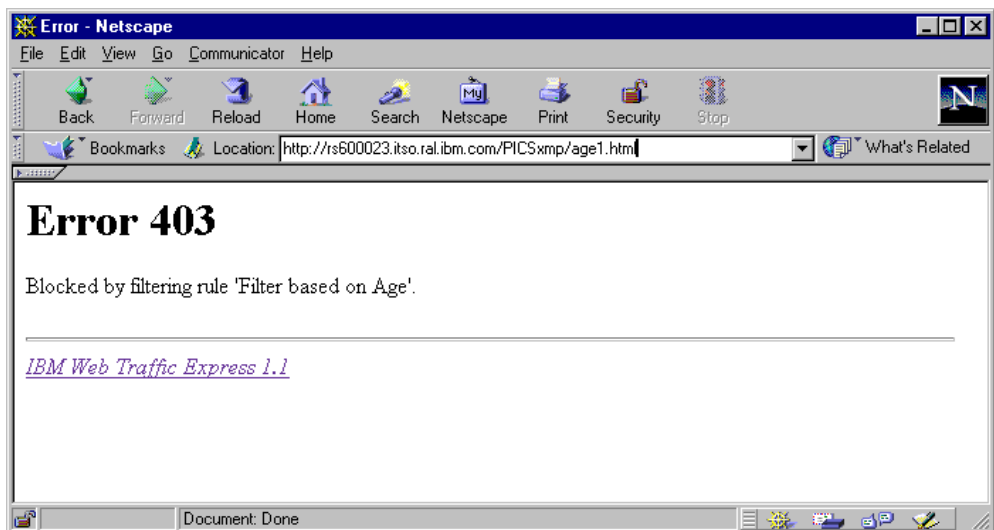


Figure 158. Another Demonstration of the failURL Directive

Then we requested the Web page `http://w3.ibm.com`, which we had explicitly permitted, and it was passed by the caching and filtering proxy server, as shown in the following figure:



Figure 159. Demonstration of the passURL Directive

The above examples demonstrated that our configuration was successful, at least for the static filters.

After that, we wanted to experiment with the conditional filters on the category `age-range` that we implemented. So we requested the URL `http://rs600030.itso.ral.ibm.com/PICSxmp/age2.html`. Since the value of category `age-range` on that specific URL was equal to 2, and we configured the caching and filtering proxy server to pass the URLs where that value was less than 3, we expected the filter rule to pass it, and in fact we got the page we requested, as shown:

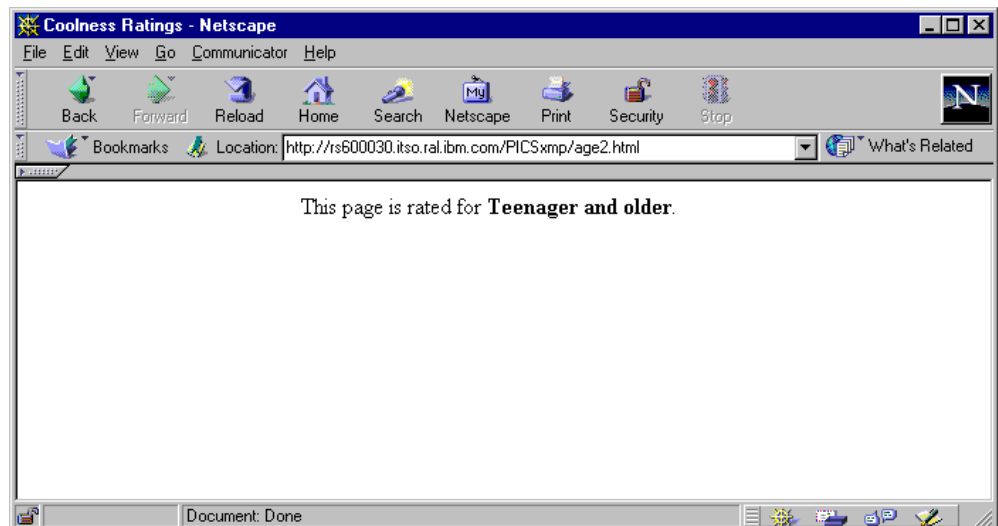


Figure 160. The Conditional Filter Passes the URL

Instead, the URL `http://rs600030.itso.ral.ibm.com/PICSxmp/age3.html` and the URL `http://rs600030.itso.ral.ibm.com/PICSxmp/age4.html` were blocked by the caching and filtering proxy server, as shown in the following two figures:

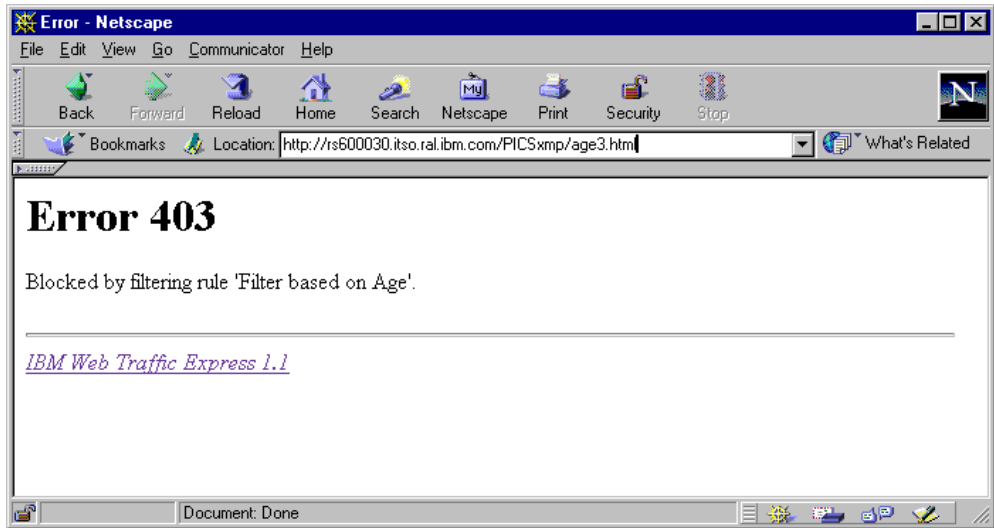


Figure 161. The Conditional Filter Rule Blocks age3.html

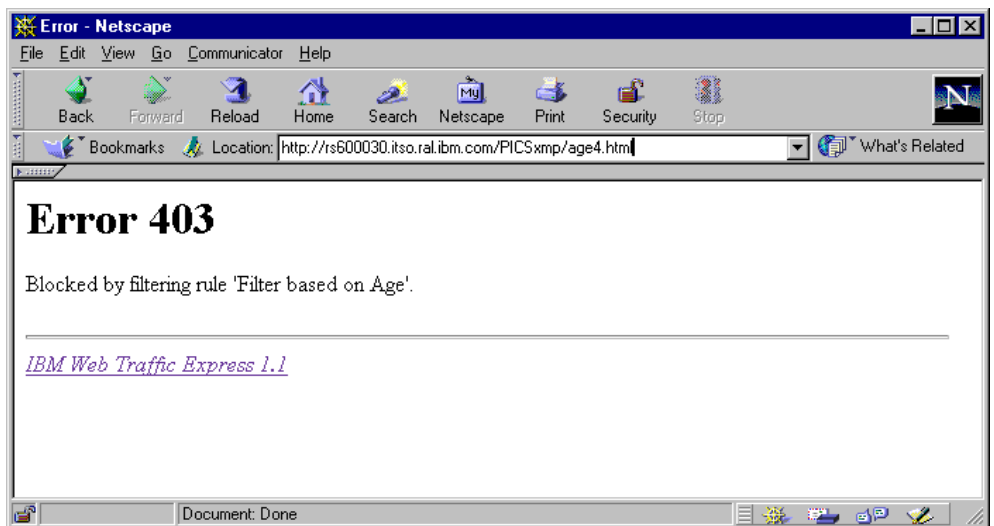


Figure 162. The Conditional Filter Rule Blocks age4.html

3.12.7 Embedding the PICS Labels in the Web Documents

The last experience we wanted to try was to eliminate the label bureau server from the PICS architecture, as shown in the following diagram:

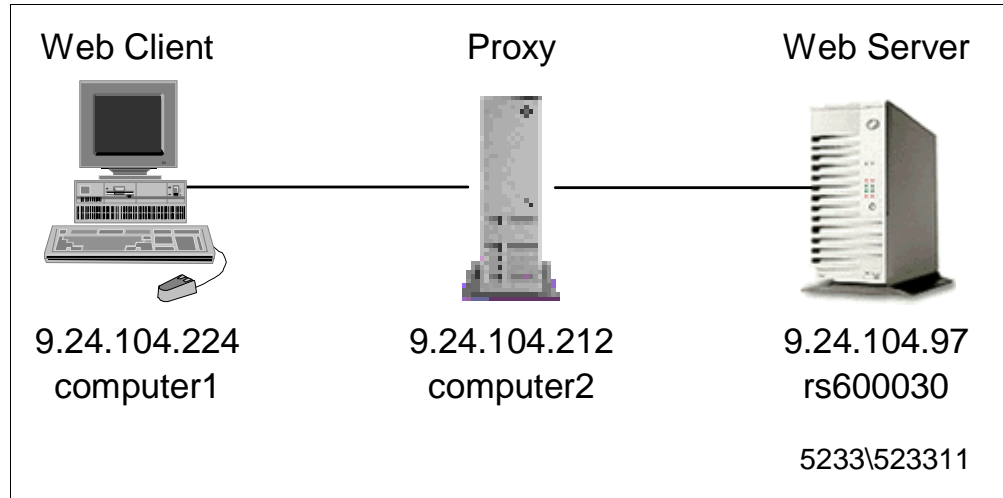


Figure 163. PICS Environment without the Label Bureau Server

We have already mentioned that it is possible, and even preferable, that when the proxy server forwards the client's request to the Web server, it is the Web server itself that provides the PICS labels to the proxy. In this way, only one connection is required, and the user on the client machine does not need to experiment with a long wait.

This operation can be performed on the server-side dynamically, by embedding the PICS labels in the HTTP headers of the Web documents while they are being sent back to the proxy (see 1 on page 182) or statically, by manually embedding the PICS labels in the HTML headers of the Web documents when these are created (see 2 on page 183). We implemented the solution to embed the PICS labels in the HTML headers.

A Web server administrator can accomplish this configuration by including the PICS labels in the `<META>` tags of the HTML document header. This method has three heavy drawbacks. First, using this method you will be able to send labels only with HTML documents, not with images, video, or anything else. Second, this process is entirely manual and therefore time-consuming, error-prone, and difficult to maintain. Third, as we have already said, it does not incorporate any of the security mechanisms (message digest, digital signature, etc.) that would guarantee the validity of the labels if this is a requirement specified on the client-side. So the static choice in general is not recommended. However, we only wanted to see what changes must be done on the caching and filtering proxy server configuration when this different approach is adopted, so we implemented this solution.

To do so, we first prepared an HTML file, called `testage4.html`, and copied it into the directory `/usr/lpp/internet/server_root/pub/PICSxmp` of the Web server machine `rs600030`. This file had to contain the PICS label in the `<META>` tag of its HTML header. The file `testage4.html` is shown in the following figure:

```

<HTML>
<HEAD>
  <META http-equiv="PICS-Label" content=
(PICS-1.1 "http://august.itso.ral.ibm.com /enzoRatings/V1-0.html"
labels on "1994.11.05T08:15-0500"
for "http://rs600030.itso.ral.ibm.com/PICSxmp/testage4.html"
ratings (Coolness 0 Age-range 4 Graphics 0)) >
</HEAD>

<BODY>
<CENTER>
<H2>The label for this page is embedded in the META tag of the header</H2>
This page is rated for <B>No one</B>.
</CENTER>
</BODY>
</HTML>

```

Figure 164. testage4.html

As you can see, the value assigned to the category `age-range` is 4.

This HTML page displayed from a Web browser would be similar to the following window:

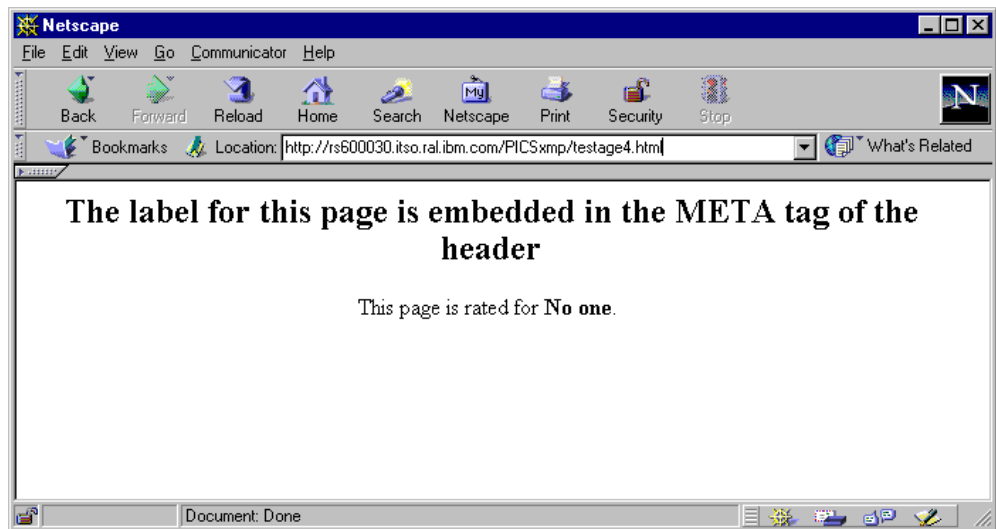


Figure 165. testage4.html Displayed on a Web Browser

Then we changed the PICS directives in the javelin.cnf configuration file of the caching and filtering server. The javelin.cnf configuration file we used in this second configuration is shown in the following figure:

```

DefinePicsRule "FilterAge" {
  (PicsRule-1.0
  (
    passURL ("http://w3.ibm.com/*")
    failURL ("http://rs600023.itso.ral.ibm.com/*")
    passURL ("http://august.itso.ral.ibm.com/Ratings")
    serviceinfo (name "http://august.itso.ral.ibm.com/enzoRatings/V1-0.html"
      shortname "TheEnzo"
      available-with-content "YES"
    )
    name (rulename "Filter based on Age - META"
      description "Fail the URL when you do not have the age")
    ibm-javelin-extensions (
      active-days "1111111"
      start-time "01:00"
      end-time "23:00"
    )
    Filter ( Pass "(TheEnzo.Age-range < 3)" )
  )
  )
}

```

Figure 166. javelin.cnf - PICS Filtering

Let's compare the above javelin.cnf file with the one we showed in Figure 156 on page 198. What differs in the above `DefinePicsRule` directive from the previous one?

In the `serviceinfo` statement we removed the information for `bureauURL` and `ratfile`, and inserted the line:

```
available-with-content "YES"
```

This informs the caching and filtering proxy server that the label is provided within the HTML file itself. No label bureau will be contacted by the caching and filtering proxy server in this case.

To distinguish this filter from the previous one, we also changed the value of the parameter `rulename`, so that the error message sent by the caching and filtering proxy server, in case of URL blocking, would contain the string

```
Filter based on Age - META
```

After that, we stopped and restarted the caching and filtering proxy server, in order for the changes in the javelin.cnf configuration file to take effect. The Web browser was still configured to send all its requests to the proxy server `computer2.itso.ral.ibm.com`. When the Web browser requested the Web server `rs600030.itso.ral.ibm.com` the HTML page `teststage4.html`, we got the error message we expected, as shown in the following figure:

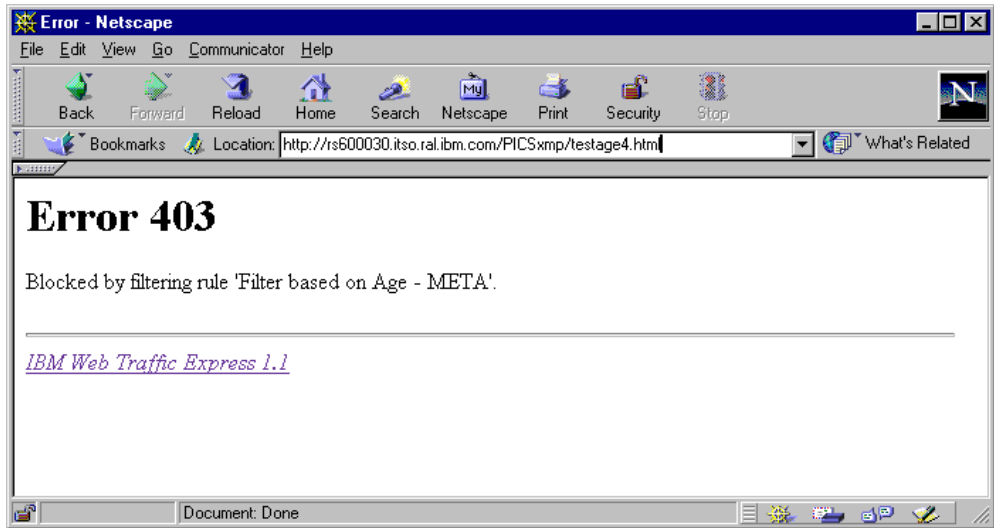


Figure 167. Error Message Displayed on the Web Browser

This error message was sent by the caching and filtering proxy server as a consequence of the filter we implemented.

Chapter 4. Load Balancing Component

This chapter gets very specific about the Load Balancing component of IBM WebSphere Performance Pack Version 1.0. This component is also known as IBM eNetwork Dispatcher Version 2.0 and it is derived from the load balancing and server monitoring software that IBM used on the Nagano Olympic Web site, where it handled peak of 104,000 hits per minute.

The first generation of Web traffic load balancing, called round-robin DNS, simply had a Domain Name System (DNS) name server rotating the resolution of names among a hard-coded list of IP addresses for Web application servers. Some degree of balancing is achieved with round-robin, and the multi-server hardware is used, but the balancing is hardly optimal, since the DNS round-robin treats all the requests as equal. Round-robin load balancing does not take into account the availability of the servers and the workload on them. Moreover DNS round-robin does not provide the ability to differentiate by port.

The Load Balancing component of IBM WebSphere Performance Pack is the next generation of server load balancing software. It improves the performance of servers since it is able to direct TCP/IP session requests to different servers belonging to a group of servers. This load balancing operation is transparent to end users and other applications.

The Load Balancing component of IBM WebSphere Performance Pack is very useful for applications such as e-mail servers, World Wide Web (WWW) servers, distributed parallel database queries, and other TCP/IP applications. When used with Web servers, it can help maximize the potential of a Web site by providing a powerful, flexible, and scalable solution to peak-demand problems. If visitors to your site can't get through at times of greatest demand, the Load Balancing component of IBM WebSphere Performance Pack can automatically find the optimal server to handle incoming requests, thus enhancing your customers' satisfaction and your profitability.

The Load Balancing component of IBM WebSphere Performance Pack consists of two functions that can be used separately or together to provide superior load-balancing results: the Dispatcher and Interactive Session Support (ISS).

- You can use the Dispatcher function by itself to balance the load on servers within a local area network (LAN) or wide area network (WAN) using a number of weights and measurements that are dynamically set by Dispatcher. This function provides load balancing at a level of specific services, such as HTTP, FTP, SSL, NNTP, POP3, SMTP and Telnet. It does not use a DNS server to map domain names to IP addresses.
- You can use the ISS function by itself to balance the load on servers within a local or wide area network using a DNS round-robin approach or a more advanced user-specified approach. Load balancing is performed at the machine level. ISS can also be used to provide server load information to a Dispatcher machine.

When used for load balancing, ISS works in conjunction with the DNS server to map DNS names of ISS services to IP addresses. When used to provide server load information, a DNS is not required.

- Using either the wide area function of the Dispatcher or a combination of the Dispatcher and ISS allows you to balance the load on servers within both local and remote networks.

We see more details in 4.1, “Functions of the Load Balancing Component” on page 210.

The Load Balancing component of IBM WebSphere Performance Pack includes the following features:

- Rule-based Web traffic load balancing
- Load balancing through firewalls to remote locations
- Integration of HTTP, mail service, FTP, Telnet and news traffic load balancing
- Independent network return paths to enhance application server responses to clients
- Agents that give live multi-server monitoring feedback
- High availability fail-over schemes

4.1 Functions of the Load Balancing Component

The two functions of the Load Balancing component are the Dispatcher and Interactive Session Support (ISS). The Load Balancing component gives you the flexibility of using these functions separately or together depending on your site configuration. This section gives an overview of the Dispatcher and ISS functions.

4.1.1 Dispatcher

The Dispatcher function does not use a DNS for load balancing. It balances traffic among your servers through a unique combination of load balancing and management software. The Dispatcher can also detect a failed server and forward traffic around it.

All client requests sent to the Dispatcher machine are directed to the server selected by the Dispatcher as optimal according to certain dynamically set weights. You can use the default values for those weights or change the values during the configuration process.

The Dispatcher has two important features:

1. The server sends a response back to the client without any involvement of the Dispatcher.
2. No additional code is required on your servers to communicate with the Dispatcher.

The Dispatcher function is the key to stable, efficient management of a large, scalable network of servers. With the Dispatcher, you can link many individual servers into what appears to be a single, virtual server. Your site thus appears as a single IP address to the world. Dispatcher functions independently of a DNS in that all requests are sent to the IP address of the Dispatcher machine.

The Dispatcher brings distinct advantages in balancing traffic load to clustered servers, resulting in stable and efficient management of your site.

4.1.1.1 Dispatcher High Availability

The Dispatcher offers a built-in high availability feature. This feature involves the use of a second Dispatcher machine that monitors the main, or primary, machine and stands by to take over the task of load balancing should the primary machine fail at any time.

For further details, see 4.11, “Dispatcher High Availability” on page 313, 4.12, “Dispatcher High Availability Scenario” on page 314 and 4.13, “Firewall High Availability Using the Load Balancing Component” on page 338.

4.1.2 Interactive Session Support

You can use the ISS function with or without a DNS name server:

- If you are using ISS for load balancing, a DNS server is required. This can either be an actual DNS server or, if you set up a small, separate subdomain for a new name server, a replacement name server provided by ISS. Using this approach, ISS runs on a DNS machine. A client submits a request for resolutions of the DNS name for an ISS-associated service, which has been set up by an administrator. ISS then resolves the name to the IP address of a server in the cell, and forwards this IP address to the client.
- If you are using ISS to collect server load information, a DNS is not needed. The ISS monitor collects server load information from the ISS agents running on the individual servers and forwards it to the Dispatcher. The Dispatcher uses this load information, along with other sources of information, to perform load balancing.

ISS periodically monitors the level of activity on a group of servers and detects which server is the least heavily loaded. It can also detect a failed server and forward traffic around it. Once every monitoring period, ISS ensures that the information used by the DNS server or the Dispatcher accurately reflects the load on the servers. The load is a measure of how hard each server is working. The system administrator controls both the type of measurement used to measure the load and the length of the load monitoring period. You can configure ISS to suit your environment, taking into account such factors as frequency of access, the total number of users, and types of access (for example, short queries, long-running queries, or CPU-intensive loads).

4.1.2.1 ISS High Availability

All the nodes in a site work together to eliminate any single point of failure. Should the monitor machine fail, the survivors elect a new monitor to take over automatically.

Implementing ISS high availability is very simple. Refer to 4.8.1, “ISS Configuration File” on page 295 for further details.

4.2 Why Do I Need the Load Balancing Component?

The number of users and networks connected to the global Internet is growing exponentially. This growth is causing problems of scale that can limit users’ access to popular sites. Currently, network administrators are using numerous methods to try to maximize access. Some of these methods allow users to choose a different server at random if an earlier choice is slow or not responding. This approach is cumbersome, annoying, and inefficient.

Another method is standard round-robin, in which the domain name server selects servers in turn to handle requests. This approach is better, but still inefficient because it blindly forwards traffic without any consideration of the server workload. In addition, even if a server fails, requests continue to be sent to it.

The need for a more powerful solution has resulted in eNetwork Dispatcher. It offers numerous benefits over earlier and competing solutions:

- Scalability

As the number of client requests increases, you can add servers dynamically, providing support for tens of millions of requests per day, on tens or even hundreds of servers.

- Efficient use of equipment

Load balancing ensures that each group of servers makes optimum use of its hardware by minimizing the hot-spots that frequently occur with a standard round-robin method.

- Easy integration

eNetwork Dispatcher uses standard TCP/IP protocols. You can add it to your existing network without making any physical changes to the network. It is simple to install and configure.

- Low overhead

eNetwork Dispatcher needs only to look at the inbound client-to-server flows. It does not need to see the outbound server-to-client flows. This significantly reduces its impact on the application compared with other approaches and can result in improved network performance.

- Non-invasive technology

eNetwork Dispatcher does not modify any packets, nor does it require any modifications to the operating system on which it runs.

- High Availability

eNetwork Dispatcher offers built-in high availability, utilizing a standby machine that remains ready at all times to take over load balancing should the primary Dispatcher machine fail.

- Server Affinity

This feature is also known as the *sticky* option. It is provided to allow load balancing for those applications that preserve some kind of state between connections on behalf of clients, without losing the state when the client reconnects. When a client originally contacts the site, it is associated with a server, chosen in the normal way. When you configure the sticky option, the difference is that any subsequent connections sent by the client will be dispatched to the same server until a configurable time-out value expires. The Dispatcher lets you to configure the sticky option on a per-port basis.

- Co-location option

The Load Balancing component can be installed on the same machine where one of the application servers reside. This option is particularly useful, especially if you want your Web site to benefit from the high availability and scalability options of the Load Balancing component with a minimum investment.

As we said, IBM WebSphere Performance Pack Version 1.0 included Version 2.0 of the Load Balancing component, also known as eNetwork Dispatcher. This version has some new features that were not present in earlier versions:

- Wide area support

With the Dispatcher's wide area support, clustered servers can now be on a different subnet from the Dispatcher, making local and remote servers possible.

- Rule-based load balancing

This function allows customers to use administrative site policies to determine how load balancing is performed (time of day, server thresholds, site-load thresholds, and so forth).

- Graphical user interface

A browser-based graphical user interface (GUI) for configuration and administration has been added, which can be used in place of the command line interface of previous versions. The command line interface can still be used as desired. A browser-based GUI interface for statistics and reporting has also been added.

- Customizable Advisors

With this version, you are given the ability to create your own Advisors, to report on a wide range of server statuses. A Java source sample is made available, with instructions for modifying it to suit the desired application. In this way, a new Advisor can be implemented quickly.

- SNMP enablement

The Load Balancing component of IBM WebSphere Performance Pack provides basic statistics and potential alert situations to allow management applications, based on the Simple Network Management Protocol (SNMP), to monitor the status of the Dispatcher.

4.3 How the Dispatcher Function Works

The Dispatcher creates the illusion of having just one server by grouping systems together into a cluster that behaves as a single, virtual server. The service provided is no longer tied to a specific server system; so you can add or remove systems from the cluster, or shut down systems for maintenance, while maintaining continuous service for your clients. The balanced traffic among servers seems for the end users to be a single, virtual server. The site thus appears as a single IP address to the world. All requests are sent to the IP address of the Dispatcher machine, which decides for each client request which server is the best one to accept requests, according to certain dynamically set weights. The Dispatcher routes the clients' request to the selected server, and then the server responds directly to the client without any further involvement of the Dispatcher. The Dispatcher can also detect a failed server and route traffic around it.

The Dispatcher receives the packets sent to the cluster. These packets have a source and a destination address; the destination address is the IP address of the cluster. All servers in the cluster and in the Dispatcher system have their own IP address and an alias for the IP address of the cluster. The Dispatcher system checks which server is the next best server to handle the load and routes the

packet to that server. Since all servers in the cluster have an alias for the cluster's IP address, the Dispatcher routes this request based on the hardware address of the network adapter (MAC address) of the chosen server. It changes the hardware address of the packet to the hardware address of the selected server and sends the packet to the server. The server receives the packet and responds directly to the client. We see a more detailed explanation of this process in 4.6.5, "How the Dispatcher Works: The Flow of the IP Packets" on page 273.

The fact that the server can respond directly to the client makes it possible to have a small bandwidth network for incoming traffic, such as Ethernet or token-ring, and a large bandwidth network for outgoing traffic, such as Asynchronous Transfer Mode (ATM) or Fiber Distributed Data Interface (FDDI).

The server machines in the cluster could be a mixture of heterogeneous servers of different sizes and types, such as UNIX, Windows NT or OS/2 machines. We see more details in 4.6, "Load Balancing Basic Scenario Scenario Using the Dispatcher" on page 246, 4.11, "Dispatcher High Availability" on page 313 and 4.9, "Rule-Based Load Balancing" on page 300.

4.3.1 Dispatcher Components

The Dispatcher consists of three main components:

1. Executor

This component supports port-based routing of TCP and UDP connections to servers based on the type of request received (for example HTTP, FTP or SSL). This module always runs when the Dispatcher function is being used.

For further details, see 4.6.3.4, "Starting the Executor" on page 252 and 4.6.3.5, "Configuring the Executor" on page 253.

2. Manager

This component sets weights used by the Executor based on internal counters in the Executor itself and feedback from the Advisors and ISS monitoring (if ISS is used as a monitoring tool). Each unit of information given to the Manager by the Advisors, ISS and Executor factor has a relative proportion of importance; so you can give more importance to one unit of information over the others, or totally ignore one or more units of information. Using the Manager is optional, but if the Manager is not used, load balancing is performed using weighted, round-robin scheduling based on the current server weights.

For further details, see 4.6.8.1, "Activating the Manager" on page 282 and 4.6.9, "Customization of the Manager and the Advisors" on page 285.

3. Advisors

Advisors send requests to the back-end servers to measure actual client response time for a particular protocol. These results are then fed to the Manager to adjust the load balancing weights. Currently, there are Advisors available for HTTP, FTP, SSL, SMTP, NNTP, POP3 and Telnet. Using the Advisors is optional, but recommended. You also have the option of writing your own Advisors.

For further details, see 4.6.8.2, "Activating the Advisors" on page 282 and 4.6.9, "Customization of the Manager and the Advisors" on page 285.

There is also an SNMP subagent component that allows an SNMP-based management application to monitor the status of the Dispatcher.

4.3.1.1 Customizable Advisors

The Load Balancing component of IBM WebSphere Performance Pack offers you the possibility to write customizable advisors that will provide the precise informations about servers that you need. Advisors, like the rest of the Dispatcher, must be compiled with Java 1.1. To ensure access to Dispatcher classes, make sure that the `ibmnd.jar` file (located in the `lib` subdirectory of the base directory) is included in the classpath system environment variable. Only advisors written to the correct level of Java will be supported.

A guideline on how to write customizable advisors, along with a sample Advisor, can be found in *eNetwork Dispatcher for Solaris, Windows NT and AIX User's Guide*, GC31-8496.

4.3.2 Proportions of Importance

The Manager decides which is the least-loaded server on a particular port in the cluster by looking at the weight of each server. The Manager will periodically update the weight of each of the server machines basing its decision on four parameters or policies:

1. The number of active connections on each TCP server
2. The number of new connections for each TCP server
3. Input from TCP server Advisors
4. Information from system monitoring tools, such as ISS

Setting the servers' weights in the load-balancing process is performed by using the so-called *proportions of importance*. Each of the above factors is attributed a number, from 0 to 100, that acts as a percentage. 0 means that the policy is not used, while 100 means that only that factor will be used. It is necessary that those proportions add up to 100. The default settings at the startup appears as 50 50 0 0.

4.3.2.1 Guidelines on Proportions of Importance Settings

Although generally there are not fixed rules on how to set the proportion value, it is still possible to provide some useful guidelines, described in the following.

The first two proportions are related to active and new connections respectively (see 1 on page 215 and 2 on page 215). Typically in an out-of-the-box configuration, this is all you will be able to use.

That is why the default proportions at startup of Dispatcher appear as 50 50 0 0. Anyway, the load on a classical Web server depends mainly on the number of connections, both active and new. Just consider the following. If the client connections to the services provided by the TCP server machines are quick (such as small Web pages served using the HTTP GET method), then the number of active connections will be expected to be fairly low. On the contrary, if the client connections are slower (such as database queries), then the number of active connections will be higher.

If you then start the Manager, it makes sense now to use a non-zero value for the Advisor proportion (see 3 on page 215). The standard Advisors shipped with

Dispatcher execute a trivial transaction on each TCP server. Normally you would expect a trivial response time to this transaction. Experience shows us that it is not a good idea to set the Advisor proportion to a high value. We recommend 49 49 2 0 for this setup.

Things change if you start using a custom Advisor. Your custom Advisor can, for example, execute a Java servlet (or other application transaction) on the server side, and this servlet can gather very precise values about performance, throughput and response time via the server's API and feed it to the advisor when asked. Thus, you may find it appropriate to put the Advisor proportion at a higher value. However, we recommend that you do not over-compensate, and do not set active and new connection proportions too low, otherwise you begin to restrict Dispatcher's adaptive and smoothing capabilities. Moreover, you may end up with a load balancer that follows your Advisor too closely, resulting in a choppy profile that is probably less than ideal. In the final analysis, what works for you is best; you need to experiment with your custom Advisor to see what results you get in the real world.

Finally, if you install the ISS daemon on your servers, it makes sense to add the ISS proportion to your mix (see 4 on page 215). If you use a custom metric in ISS as well as a custom Advisor, be sure to have them go after similar objectives, otherwise you may come to set one against the other with undesirable consequences.

4.3.3 Information Flow

The typical flow of information used in the Dispatcher is shown in the following diagram:

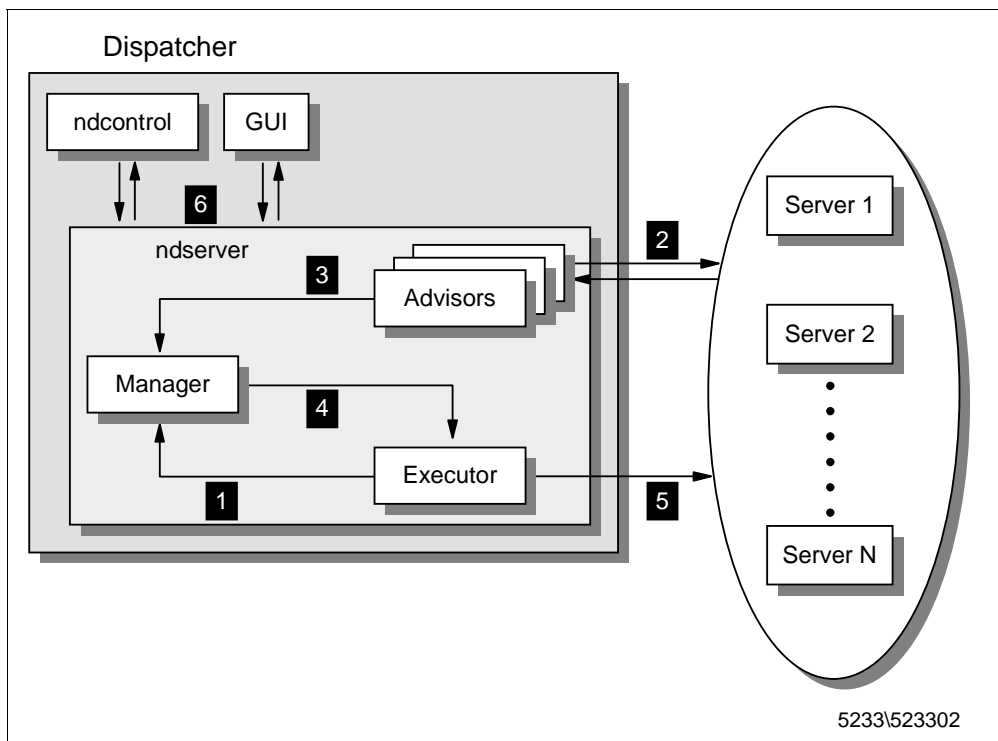


Figure 168. Information Flow Used in the Dispatcher

The first step that must be done after installing and configuring the Dispatcher is to run it using the following command:

```
ndserver start
```

The logical information flow, represented in the above diagram, is described in the following list:

1. The Executor supplies information about new and active connections based on its internal counters to the Manager.
2. The Advisors collect information about response time and availability from the servers for each service and each port.
3. After processing this information, they send it to the Manager.
4. The Manager uses all the information it receives (including information sent by the ISS component, if active, about the servers' load balancing according to a specific metric), calculates the new weights to be used in connection routing and sends the results to the Executor.
5. The Executor uses these new weights for TCP and UDP routing. If the Manager and the Advisors are not running, the Executor does the routing based on its internal counters.
6. A command line interface, represented by the `ndcontrol` command, and a graphical user interface (GUI) are provided to configure and manage the Executor, Advisors and the Manager.

4.3.4 TCP Ports Used by the Dispatcher

The Dispatcher uses three TCP ports for its communications:

1. Port 1099

This port is used for GUI communications.

2. Port 10003

This port is used for receiving commands.

3. Port 10004

This port is used to receive metric response from ISS.

Sometimes it is necessary to use port numbers other than the defaults. If another application is already using port 1009 or port 10003 for communication, then you will need to change the `ND_PORT` value in the Dispatcher scripts to use different ports:

- On AIX, these scripts are named `ndadmin`, `ndserver` and `ndcontrol` and are located by default under the `/usr/bin` directory.
- On Windows NT, these scripts are named `ndadmin.cmd`, `ndserver.cmd` and `ndcontrol.cmd` and are located by default under the `Bin` subdirectory of the Load Balancing component `eND` directory.

To change the port used to receive metric reports from ISS, use the `metric_port` option when starting the manager:

```
ndcontrol manager start log_file metric_port
```

Notice that if you specify a *metric_port* you must also specify a *log_file*. The above command has a correspondent version in the GUI, as we see in 4.6.8, “Activating the Manager and the Advisors” on page 282.

Moreover, when you define the Dispatcher Observer in the ISS configuration file, you should indicate the port you have decided to use:

```
Dispatcher hostname metric_port
```

This is shown in Figure 245 on page 297.

4.4 How the ISS Function Works

This section describes the ISS function and its concepts.

4.4.1 ISS Cells and Services

ISS logical architecture is based on the concept of a *cell*. A cell is a group of servers administered as a single, logical unit. Each server in a cell is also called a *node*.

A group of servers (or nodes) that perform the same function is a *service*. Think of a service as a group of nodes in the cell that will serve the request only for a particular protocol (for example HTTP, FTP, Telnet).

Is it possible in a cell to define different services (corresponding, for example, to HTTP, FTP, Telnet). Moreover a node can belong to one or more services. The following figure offers a graphical representation of a possible situation:

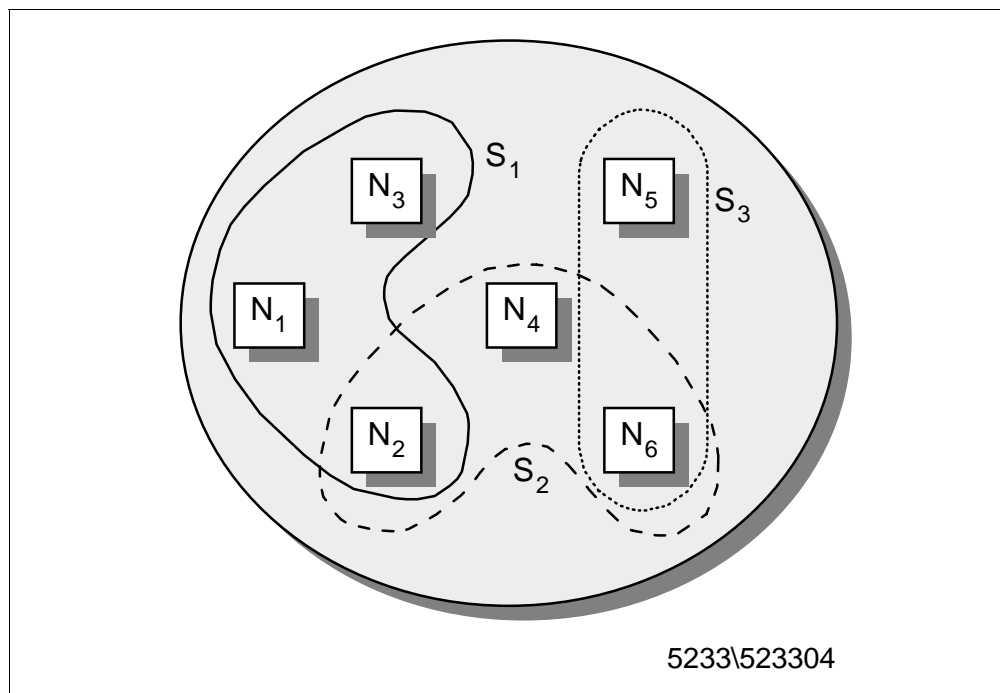


Figure 169. Cell, Services and Nodes

In this example, we have a cell with six nodes, named N1, N2, N3, N4, N5 and N6, and three services, S1, S2, S3. You can see that:

- The service S1 contains the nodes N1, N2 and N3.
- The service S2 contains the nodes N2, N4 and N6.
- The service S3 contains the nodes N5 and N6.

Note that some nodes belong to only one service, and other nodes belong to more services. In particular:

- N1 and N3 run only service S1.
- N4 runs only service S2.
- N5 runs only service S3.
- N2 runs services S1 and S2.
- N6 runs services S2 and S3.

On each node of the cell the ISS component must be installed, configured and activated. The ISS will run as a daemon, called the `issd` daemon.

4.4.2 ISS Configuration Files

When configuring ISS, you need to edit a configuration file that instructs the `issd` daemon on how to act. Examples of configuration files are shipped with the product. In fact, the installation of the Load Balancing component of IBM WebSphere Performance Pack comes with three ISS sample configuration files: `Iss_config_1`, `Iss_config_2` and `Iss_config_3`. These three ISS sample configuration files are very useful for understanding how to configure ISS in your environment:

1. The file `Iss_config_1` shows the configuration of a local cell. It is a simple configuration file with only one local cell and one service running.
2. The file `Iss_config_2` illustrates how ISS can be used with a DNS name to perform load balancing for three clusters of servers. A different load balancing method (round-robin, custom and LoadLeveler) is used for each cluster.
3. The file `Iss_config_3` illustrates how ISS can be used with the Dispatcher in a two-tiered architecture. The bottom tier consists of three clusters of servers. Each cluster itself has two servers. Three Dispatchers are used to balance these clusters. The top tier consists of three Dispatcher nodes. In this configuration, ISS performs two functions:
 1. It performs DNS load balancing for the top tier.
 2. It provides the Dispatchers with additional server load information, so that they can load balance the bottom tier more accurately.

We recommend that you open those configuration files with a text editor and see how they are structured. If you selected the U.S. English language, the three ISS sample configuration files will be located:

- In the directory `/usr/lpp/eND/iss/samples/en_US` on AIX
- In the directory `iss\Bin\samples\en_US` under the Load Balancing directory on Windows NT

In the following figure, we show you the ISS sample configuration file `Iss_config_1`, since it will help you understand what we say in the next sections about ISS. Moreover, in 4.8.1, "ISS Configuration File" on page 295, we show

how we modified that file to implement a load balancing scenario making use of both Dispatcher and ISS.

```

# -----
# Sample ISS configuration file 1
# -----
#
# Configuration of a local cell
# This is a simple configuration file,
# with only one (local) cell, and one service
# running.
# Parameters for the whole cell
Cell      Hursley          local
AuthKey   10043572 ADE4F354 7298FAE3 1928DF54 12345678
LogLevel  info
HeartbeatInterval 5
HeartbeatsPerUpdate 3
PortNumber 7139

# Individual node data
# Node numbers do not have to be sequential
# nemesis is prevented from taking over the role
# of monitor.
Node  delta          001
Node  spcontrol      002
Node  spnode1        003
Node  spnode2        004
Node  spnode3        005
Node  spnode4        006
Node  spnode5        007
Node  nemesis        099
NotMonitor

# The service is only configured to depend on
# one resource -- CPU availability.
# Load balancing is therefore performed based
# only on CPU utilisation. However, ISS will
# not schedule work for nodes that are unreachable
# on the network.
# The specified MetricLimits indicate that a node
# will not be used if its CPU usage goes over 95%
# and will not be put back in the list until CPU usage
# goes back down to 80%.

ResourceType      CPU
Metric Internal   CPUload
MetricNormalization 0 100
MetricLimits      80 95
Policy Min

# The one configured service
# WWWService is an identifying string for the service, and
# is used when balancing between cells.
# The service DNS name is www.issdev.hursley.ibm.com.
# followed by the cluster address and the port number.

```

Figure 170. (Part 1 of 2). Iss_config_1 Sample Configuration File

```

Service      WWWService      www.issdev.hursley.ibm.com 129.40.161.74 21
NodeList
ResourceList      CPU
SelectionMethod  Best
Overflow          spnode5.hursley.ibm.com

# The machine 'delta.hursley.ibm.com' is configured as
# an ISS Nameserver -- when the issd program starts on
# machine delta, it will also try and run an ISS name
# server thread on port 53.
ISSNameserver    delta    53
ServiceList     WWWService

# The machine nemesis.hursley.ibm.com has a
# Network Dispatcher configured on port 10004
Dispatcher      nemesis 10004
ServiceList     WWWService

```

Figure 171. (Part 2 of 2). *Iss_config_1 Sample Configuration File*

In the following sections, we see how the ISS function works to understand the settings shown in the above configuration file better. We recommend that you read the information provided in the following sections in parallel with the above configuration file.

4.4.3 Cell and Its Attributes

The first thing that an ISS configuration file does is to define the cell, and declare some cell attributes.

A cell can be local or global, and this is specified in the ISS configuration file through the keywords `local` or `global` respectively. A local cell is the one that the node will be a member of. There must be one and only one local cell defined in the ISS configuration file. A global cell is a separate, remote cell that you want your node to communicate with. You must have a global cell if you have widely separated mirror sites and want to perform *ping triangulation*, which is a technology that allows you to find out which server site is the closest to a given client.

A cell can have the following attributes, specified in the ISS configuration file:

- `Authkey`

This key is used to provide authentication during the ISS internal traffic among the nodes of the cell. If this keyword is not specified in the configuration file, a default key will be used. A node can have its own `Authkey` attribute, which has to be specified after the node is defined.

- `LogLevel`

You can have five different log levels, each one providing more information than the previous one:

```
[ None | Error | Info | Trace | Debug ]
```

The above values for the `LogLevel` attribute are self-explanatory.

- `PortNumber`

This option specifies the port number used for ISS internal traffic:

```
PortNumber [number]
```

- HeartbeatInterval

ISS will periodically contact all servers that provide a specific service to ensure that they are still alive. ISS does this using the equivalent of the `ping` command. This does not verify that the service is running on the servers, but at least ensures that the servers can be contacted at the IP level. ISS maintains a ranking that lists the servers in their current priority order according to the configured load measurement metric. If the currently top-ranked server fails to respond, the next server will be selected. This check is referred to as a heartbeat. The time between heartbeats is determined by the `HeartbeatInterval` setting. The syntax is:

```
HeartbeatInterval [seconds]
```

- HeartbeatsPerUpdate

The syntax of this keyword is:

```
HeartbeatsPerUpdate count
```

Where *count* is a non-negative integer.

The `HeartbeatsPerUpdate` value is multiplied by the value of `HeartbeatInterval` to give a time called the *update interval*. The value assigned to the update interval parameter indicates the number of heartbeats after which ISS will determine whether a domain name server update or a Dispatcher update is required.

Note that the value of `HeartbeatsPerUpdate` will be ignored in the situation where the top ranked server fails to respond to a heartbeat. The server will immediately be removed from its top ranked position.

4.4.4 Nodes

After defining the cell, you must list all the nodes that will be part of it using the `Node` keyword. The syntax is:

```
Node [ hostname | IP address ] priority
```

In your cell configuration, when using ISS, you should elect one of the nodes as the cell leader, or *monitor*, while all the other nodes act as *agents*. In other words, if ISS is used to collect server load information, there is an ISS monitor that collects this information from the ISS agents that run on the individual servers. The ISS monitor is then used to send the ISS agent information to the Dispatcher. The ISS monitor can be installed on one of the servers, a Dispatcher machine or on a different machine.

Moreover, you can configure one or more other nodes as backup of the monitor node, by defining a different priority number for each one. In a backup node, the `issd` daemon usually runs in agent mode, and the node can only be a server that provides a service. If the monitor fails, the first (in the priority scale) backup node in the list takes over, and on this node the `issd` monitor switches from agent mode to monitor mode. As soon as the previous node is available again, it will become the monitor because of its higher priority value. This functionality provides ISS *high availability*.

If you don't want a particular machine to run as an issd monitor, you have to declare this using the `NotMonitor` keyword immediately after the node declaration.

In this case the priority field is only a numeric identifier for that node.

4.4.5 Services

Before explaining how to define the resources, we prefer first to focus on the section of the configuration file where the services are defined, along with their parameters:

- `Service`

You declare a service with the `Service` keyword, followed by the service name, the fully qualified service DNS name (which is the DNS name of the pool of machines that provide the same service), the cluster IP address and the port number. The correct syntax is:

```
Service name DNSName [ cluster address ] [ port number ]
```

- `NodeList`

After defining the service, you need to use the `NodeList` keyword, which allows you to specify the list of nodes and members of your cell providing the service. `NodeList` must be followed by the hostnames or IP addresses of those servers:

```
NodeList DNSName [ DNSName ... ]
```

- `ResourceList`

The `ResourceList` parameter allows you to declare which resource will be used to determine the appropriate node that will provide that service. A resource can be CPU, disk, process, and so forth.

- `Overflow`

This keyword identifies a server on which to fall back if all the other nodes specified in `NodeList` are unable to provide the service. The syntax is:

```
Overflow DNSName
```

4.4.6 Resources

When you define a service in the configuration file, you specify the name of one or more resources as arguments of the `ResourceList` parameter. A single resource can be used as the selection criterion in different services. Consequently, the resources should be defined in the configuration file before the service definitions.

You define resources using the `ResourceType` parameter and its own keywords `Metric`, `MetricNormalization`, `MetricLimits`, and `Policy`.

The `ResourceType` keyword must be followed by the symbolic name of the resource you are defining. It specifies the criteria you have decided to use for selecting the best server in a service. The syntax is:

```
ResourceTypes Name
```

We give you now further details on how to use these parameters.

4.4.7 Metrics

A resource type is characterized by several parameters that define the metric that will be used to measure the load among the servers.

A metric defines how ISS will measure the load on the server for each service. This measurement should be appropriate to the particular service. For example, if the service provided is CPU-intensive, the metric could be defined as the percentage of time that the CPU is busy. If the service is disk- or I/O-intensive, the metric could be based on the amount of time that the disks are busy or the time spent waiting for I/O to complete.

The metric keywords are the following:

- `Metric`

The keyword `Metric` in the configuration file specifies the type of measurement ISS will use to provide a specific service. The syntax is:

```
Metric [ Internal | External ] string
```

where the *string* field indicates a command or a program that gives a numeric output.

The sample configuration file `ISS_config_1` that we have shown uses the line:

```
Metric Internal      CPULoad
```

The keyword `Internal` identifies a given measurement method as being provided by the system. The parameter `CPUload` forces ISS to measure the percentage of the CPU that is being utilized. The other system-provided measurement system is `FreeMem`, which returns the amount of free physical memory as a percentage.

When you set `Metric` to `External`, you define how to measure the load on the servers. This metric is anything that can be executed on the server and returns a numeric value as a result. In this case the *string* field defines a command or program that when executed, returns a numeric value that can be used to measure of the load on the server.

For example, the following entry defines a metric based on the number of processes running on the server:

```
Metric External ps -ef | wc -l
```

- `MetricNormalization`

This is a range between the lower and upper limits of measurement, indicated as integer numbers. The syntax is:

```
MetricNormalization LowerLimit UpperLimit
```

The limits referred to depend upon the metric you are using. In case you want to measure the CPU utilization, as shown in the `Iss_config_1` sample configuration file, the two limits are percentage points. The range is therefore 0 to 100. Any metrics coming into the monitor outside of this range would be corrected by being clipped to fit within these limits.

- `MetricLimits`

The syntax for this entry is:

```
MetricLimits RecoverLimit FailLimit
```

You can set these limits to prevent a server from failing totally by having too many requests assigned to it. ISS measures the loads on servers periodically. If the loads on a certain server are within predefined limits, ISS continues keeping the server in the list of the available nodes. If the loads are outside the limits, ISS removes the server from the ranking for that service. If the server is handling more than one service, it may still be included in the ranking for the other service or services. There are two levels you define:

1. The *FailLimit* level represents the level beyond which the metric should not go. When the *FailLimit* level is reached, ISS removes the server from ranking. In the *Iss_config_1* configuration file that we have shown, the specified value for *FailLimit* indicates that a node will not be used if its CPU usage goes over 95%.
2. The *RecoverLimit* level sets the point at which a node that had been removed from the ranking should be returned to active participation. In the *Iss_config_1* ISS sample configuration file, the *RecoverLimit* parameter is set to 80%.

So, defining the limits of the CPU load metric with the following entry causes the resource to be removed at 95% of utilization, and returned only when it is back down to 80%:

```
MetricLimits 80 95
```

Specifying a significant difference between the two above levels prevents the phenomenon wherein resources come into service, fail, then come back after minimal recovery only to quickly fail again.

- **Policy**

The parameter *Policy* can be assigned either the value *Max* or *Min*. The correct syntax is:

```
Policy { Max | Min }
```

It indicates whether a metric function is to be minimized or maximized. The default value is *Max*.

For example, a metric based on the number of active users would regard the smallest value as the best and would have the following entry:

```
Policy Min
```

A metric function based on CPU idle time on the other hand, would regard the maximum value as best and, therefore, the policy would be specified as the following:

```
Policy Max
```

4.4.8 ISS Observers

After configuring nodes and services, and after defining the resources, you should define who will use the information about the status of nodes. In other words, you should configure the observers.

An *observer* is a network process (such as a load balancer) that uses information about the status of nodes. ISS provides three observers, which perform different actions, but have the same purpose: load balancing. The three observers are named *NameServer*, *ISSNameServer* and *Dispatcher*. These names correspond to the keywords *NameServer*, *ISSNameServer* and *Dispatcher* that you should use in your ISS configuration file.

Depending on how you decide to configure your environment, you can configure one or more observers in your cell.

4.4.8.1 NameServer

When using the NameServer observer, ISS runs in conjunction with a DNS name server. It uses the DNS name server to map DNS names of ISS services to IP addresses of the most appropriate server. ISS assists the DNS server in making the balancing decision. ISS monitors the load on each server of the cell and ensures that the server currently used for a particular service is the one with the lightest load.

It is not mandatory that the machine where the `issd` monitor process runs is the same machine where the DNS daemon `named` runs. For this reason, you can configure one or more servers on your cell as backup `issd` monitor nodes.

The load balancing decisions are based on how you have configured ISS. In this case, ISS makes load balancing decisions by itself and instructs the DNS name server about the server that the incoming request are to be routed to.

The NameServer observer involves some load on the DNS server machine. Every time a new server is selected as the top ranked server, the DNS configuration files are updated. A signal is then sent to the `named` daemon to reload the name resolution data files. If these files are large, it can take a significant amount of time and processing for the `named` daemon to process them. During this processing, the `named` is unable to respond to name resolution requests.

The syntax to define a NameServer observer is the following:

```
NameServer DNSName [ PortNumber ]
```

4.4.8.2 ISSNameServer

When using the ISSNameServer observer, ISS works as a DNS name server. In this case, ISS runs on a DNS domain name server machine, where the DNS daemon `named` is not running.

ISS replaces the `named` name server daemon and makes use of the DNS configuration file of the DNS name server. In this case ISS provides the name serving function by providing minimal name server implementation. In this way, besides taking over the DNS, ISS makes load balancing decisions by itself.

Using the ISSNameServer observer does not increase the load on the DNS machine, so it is a good option to use the ISSNameServer observer with a new subdomain for your pool of servers.

The syntax to define an ISSNameServer observer is the following:

```
ISSNameServer DNSName [ PortNumber ]
```

For example, the following is the entry used in the `Iss_config_1` sample configuration file:

```
ISSNameServer delta 53
```

4.4.8.3 Dispatcher

When using a Dispatcher observer, you should have defined in your cell a Dispatcher machine, which ISS strictly cooperates with to perform load

balancing. Different from the previous two types, a DNS server is not required (neither a DNS server nor an ISSNameServer observer working as DNS).

The ISS configuration file will reflect your cell configuration, and the Dispatcher machine you selected should be specified immediately after the `Dispatcher` keyword, according to the following syntax:

```
Dispatcher DNSName [ PortNumber ]
```

In such a configuration, you can look at ISS as a monitoring tool on the TCP server machines. ISS provides the Dispatcher with load server information, but ISS does not make any load balancing decision.

The ISS monitor collects specific server information, such as CPU usage, memory usage and disk activity, from the ISS agents running on the individual servers, and forwards it to the Dispatcher. The Dispatcher uses this load information, along with other sources of information, to determine which is the least-loaded server of the cluster and then performs load balancing.

4.4.9 ISS Selection Methods

For each observer you plan to use in your cell, you should specify a selection method, which defines how the ISS monitor selects the best node to perform that service. You can use two selection methods: RoundRobin and Best, identified with the keywords `RoundRobin` and `Best` respectively.

1. `RoundRobin`

The load among servers is distributed on a round-robin basis; the load on each server is ignored, although a server will not be recommended if it appears to be down.

The main advantage of using round-robin is that round-robin avoids overhead in the server associated with other measurement types. The main disadvantage of using round-robin is that ISS does not notice whether a particular service is getting very busy, and ISS could continue to use a busy server as the server to which new users connect.

2. `Best`

For each service, the ISS monitor maintains a ranking of the best nodes to perform that service. Periodically, the monitor calculates the best server for every defined service and updates the ranking of the nodes. Before the next updating (that is, for the update period), ISS will use (or recommend to use) the highest server in the ranking.

In order to establish the ranking, ISS monitors the load on each server and selects for a particular service, the node that best performs that service. In this case, the node selection performed by ISS truly depends on the resources defined for the service.

Server Failure

For both selection methods, ISS detects if a server fails and does not schedule requests for that service.

A group of nodes performing the same service are independent of each other, and the choice of measurement type for one does not affect the choices available for the others. You can have several such groups, with each operating in a different manner. For example, you can configure a round-robin service and additionally provide a service using the `Dispatcher` observer with the `Best` selection method.

4.5 Installation of the Load Balancing Component

The Load Balancing component of IBM WebSphere Performance Pack is supported on three operating systems: IBM AIX 4.1.5 or later, Microsoft Windows NT 4.0 and Sun Solaris 2.5 or later. In this section we show you step by step how to perform the installation on AIX and Windows NT. The installation on Solaris is pretty similar to the installation on AIX. For further details you can refer to the *eNetwork Dispatcher for Solaris, Windows NT and AIX User's Guide*, GC31-8496.

4.5.1 Installation on AIX

Before starting our discussion, it would be good to describe the hardware and software environment on which we performed our installation. The machine we used as our platform was a uniprocessor IBM RS/6000 43P having 192MB of RAM, 2.2GB of hard disk and one token-ring interface. We installed this machine with AIX Version 4.3.1.

The AIX installation program for each component of IBM WebSphere Performance Pack requires that the Java Virtual Machine (JVM) Version 1.1 or later is installed on the system.

We did not need to install the JVM on our machine, because the Java Development Kit (JDK) 1.1.4 fileset is automatically installed with AIX Version 4.3.1. Notice that IBM WebSphere Performance Pack requires AIX Version 4.2.1 or later.

By entering the following command we noticed that the default installation of AIX 4.3.1 locates the JVM java executable file in the directory `/usr/bin`:

```
which java
```

The CD of IBM WebSphere Performance Pack is provided with the JDK 1.1.4 installation file for AIX, found in the directory `JDK/AIX`. You need to install the JVM if it is not already installed on your system, or you might want to upgrade the level if it is previous to 1.1.4. To discover the level of the JVM already installed on your machine, you can enter the command:

```
java -version
```

The installation of JDK on AIX is described in the IBM redbook *Network Computing Framework Component Guide*, SG24-2119.

To prepare for installation, follow the steps listed below:

1. Insert the IBM WebSphere Performance Pack CD-ROM in the CD-ROM drive.
2. From a command line, enter the following commands:

```
mkdir /CD-ROM  
mount -rv cdrfs /dev/cd0 /CD-ROM
```

3. Enter `cd /CD-ROM/AIX`.
4. To start the installation program, enter `java setup`.

After a while, you will get the Welcome window. Then, after clicking the **Next** button, you are required to agree to all the items of the software license. If you agree, check the box **Accept all terms of the license**, then click **Next**. You will see another window displaying the IBM WebSphere Performance Pack Version 1.0 Readme File. We suggest that you take a look at the file as it contains interesting information about the product. Then click **Next**, and a dialog will be displayed where you can select the language, but *only* for the Load Balancing Component:



Figure 172. Load Balancing Component Language Selection

Notice that this option is available even if you have not yet specified that you want to install the Load Balancing component. After you click **Next**, you are prompted to enter the IBM WebSphere Performance Pack destination location. We accepted the default `/public/WebSphere`, and we had the setup program create such a directory, as shown in the next window:

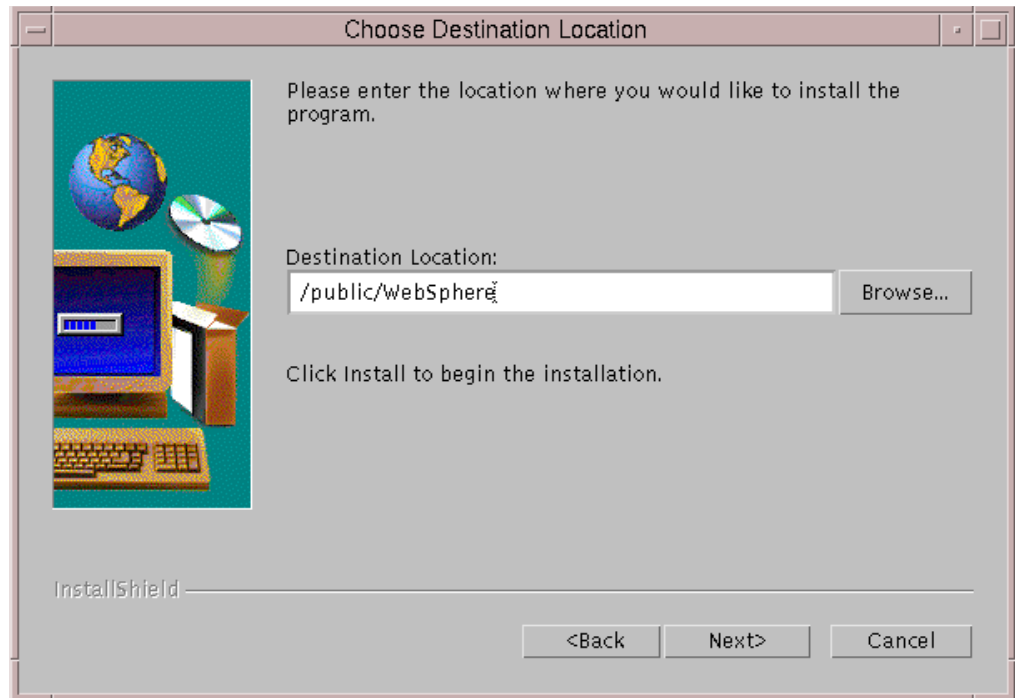


Figure 173. Choose Destination Location

In the above window, you are prompted to click **Install** to proceed. Click the button labeled **Next** to continue instead.

Select now the IBM WebSphere Performance Pack components that you want to install. Note that the installation program allows you to install on AIX these combinations of components:

- Load Balancing (eNetwork Dispatcher) component and/or Caching and Filtering component and/or File Sharing client component
- File Sharing server component and/or File Sharing client component

We selected **Load Balancing (eNetwork Dispatcher)**, and the File Sharing server check box appeared immediately disabled, as shown in the following screen:

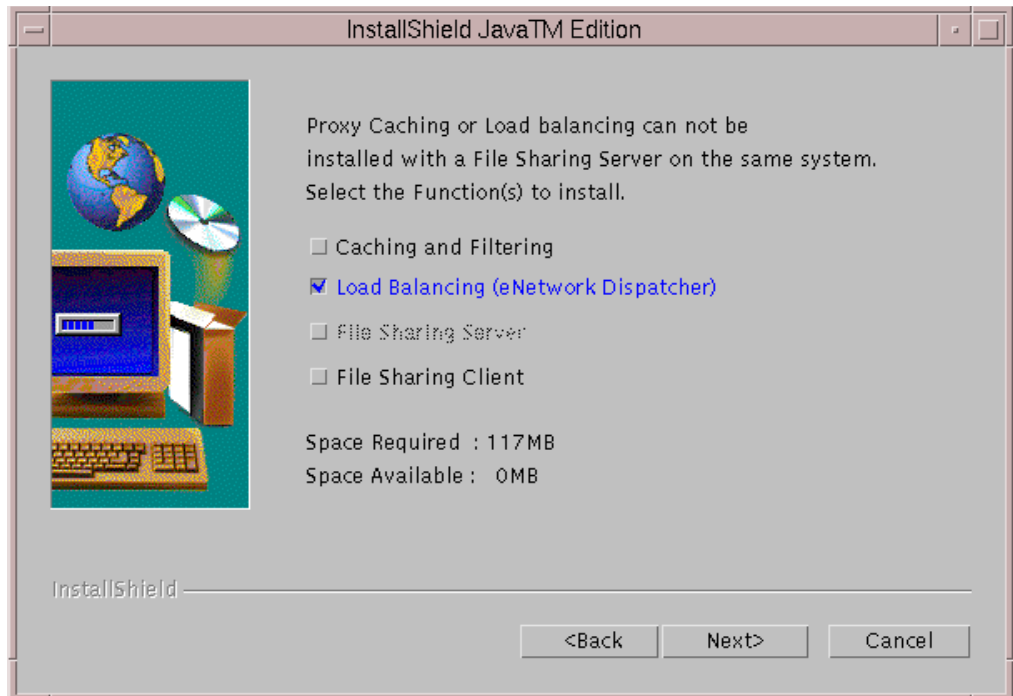


Figure 174. No Space Available for Installation

Note that we accepted the default settings to install the product in the /public/WebSphere directory. As you can see in the above window, it turned out that the available space in the / file system was 0MB. In other words, that file system was empty, and we needed to increase its space to install the Load Balancing component, which requires 117MB. For this reason, we couldn't go on with the installation. By entering the following command we received the confirmation that no free space was available in the mentioned file system:

```
df -k
```

Then we enlarged the file system size following the steps listed below:

1. From a command line, we entered `smitty jfs`.
2. We selected **Change / Show Characteristics of a Journaled File System**.
3. We chose the file system `/`.
4. Since 117MB of free space is required (see Figure 174 on page 232), we added 250,000 512-byte blocks to the size of file system (see Figure 175 on page 233), which became 270,336 512-byte blocks sized.

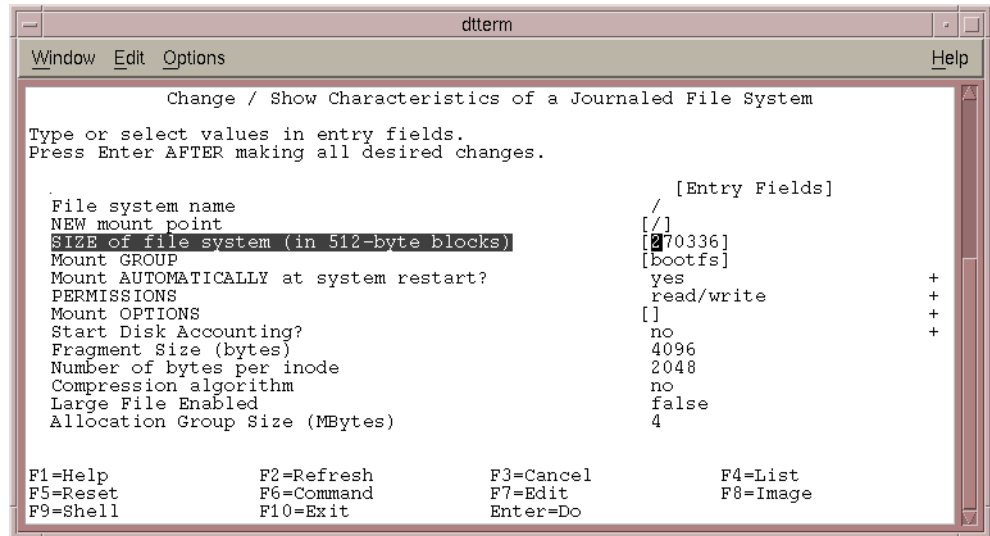


Figure 175. Enlarge the Size of the File System

After expanding the space in the file system, we had enough free space to proceed with the installation, as shown in the following window:

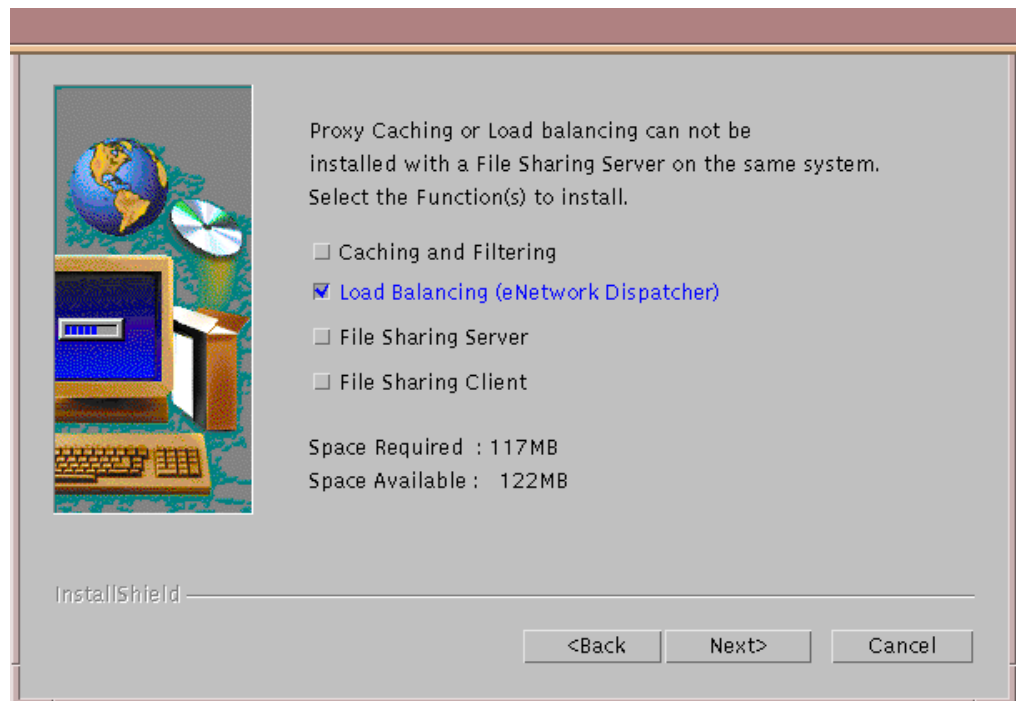


Figure 176. Free Space Enough for the Installation

After you click **Next**, you will get a window that asks if you want the installation program to replace other programs already installed on your system. We selected **No** in this case, since we had not installed any programs on our system yet.

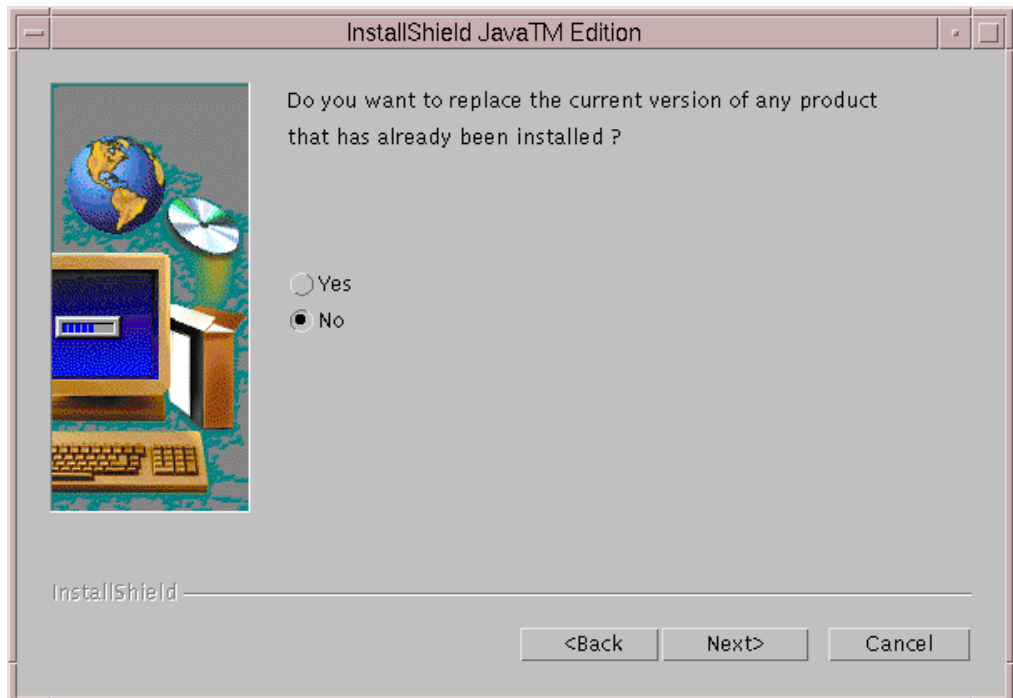


Figure 177. Choose to Replace Version

When we clicked **Next**, the following window was displayed:

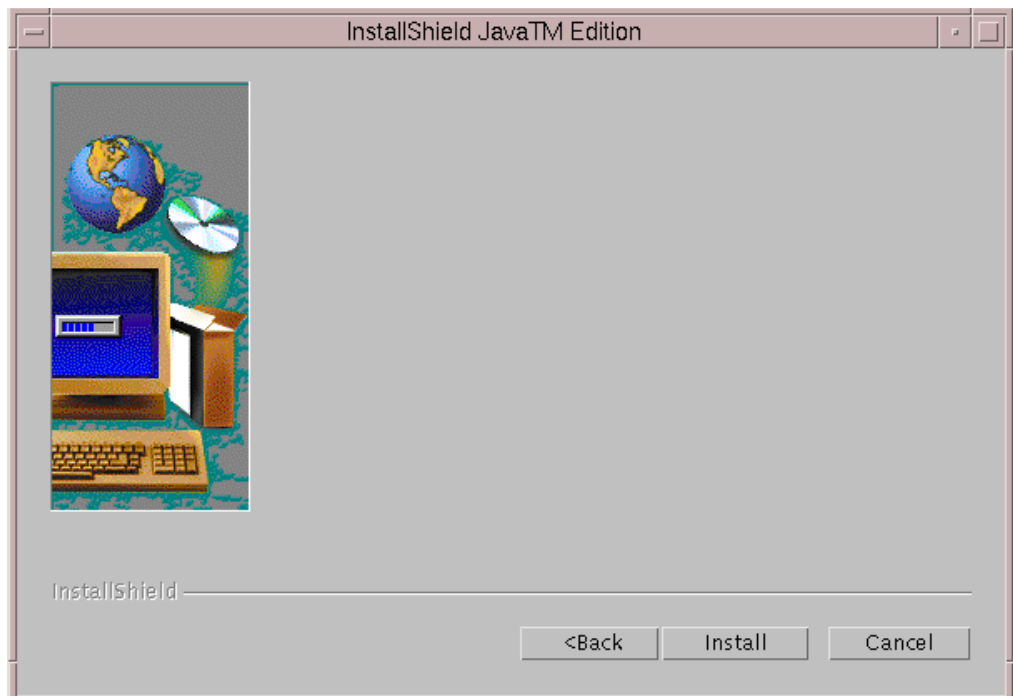


Figure 178. Start Installation Window

Click the **Install** button and after a while you will be informed that the installation is complete.

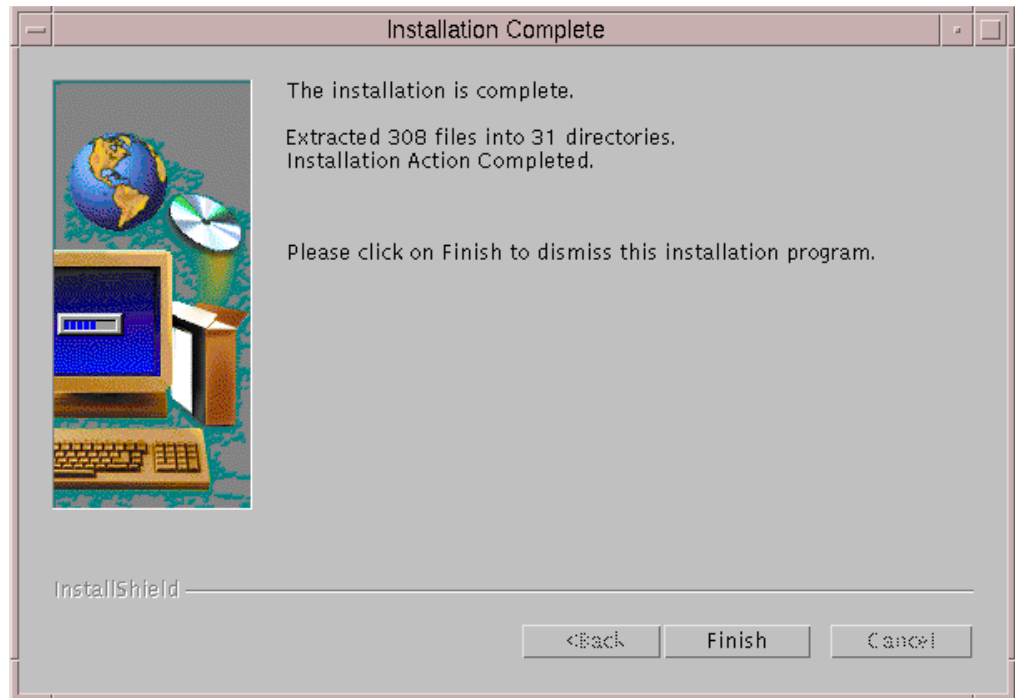


Figure 179. Installation Complete

After clicking **Finish**, we verified that the product had been installed by entering the following command:

```
lslpp -h | grep intnd
```

If the product has been correctly installed (and if you chose the U.S. English version), you will see the following:

```
intnd.iss.license  
intnd.iss.rte  
intnd.msg.en_US.iss  
intnd.msg.en_US.nd  
intnd.nd.license  
intnd.nd.rte  
intnd.ps.en_US
```

4.5.2 Installation on Windows NT

In this section we describe all the steps that were necessary to install the Load Balancing component of IBM WebSphere Performance Pack on our Windows NT platform.

Before starting with the installation, it would be good to describe the hardware and software environment on which we performed this installation. The machine was an IBM PC 750 with 166 MHz of CPU, 96MB of RAM, 1.5GB of hard disk and one token-ring adapter. This machine had been installed with Windows NT Server 4.0 and Service Pack 3.

The Windows NT installation program for each component of IBM WebSphere Performance Pack makes use of Java InstallShield's setup class.

For this reason you are required to pre-install the JVM Version 1.1 or higher, which is incorporated into the JDK Version 1.1 or higher. Actually you wouldn't need the full JDK, but only its subset known as Java Runtime Environment (JRE), which contains just the JVM, the Java platform core classes, and supporting files. In other words, the JRE is the smallest set of executables and files that constitute the standard Java platform and it contains only the run-time part of the JDK: no compiler, no debugger, no tools. The CD of IBM WebSphere Performance Pack ships with the JDK 1.1.5 for Windows NT, found in the directory JDK\NT; however, we preferred to install the JDK 1.1.6 for Windows NT, since that was the latest non-beta version that was available at the time we were writing this redbook.

The latest version of the JDK can be downloaded for free from the JavaSoft Web site <http://www.javasoft.com>.

The JDK 1.1.6 installation for Windows NT is very easy and so we skip its description. However, for further details, you can see the IBM redbook *Internet Security in the Network Computing Framework*, SG24-5220.

To install the Load Balancing component of IBM WebSphere Performance Pack, also known as eNetwork Dispatcher (eND), run the setup.exe program from the CD-ROM installation directory, named NT. To do this, from the Start menu, select **Run...**, then click on **Browse...** and open the setup.exe program located in the NT directory, as shown in the following figure:

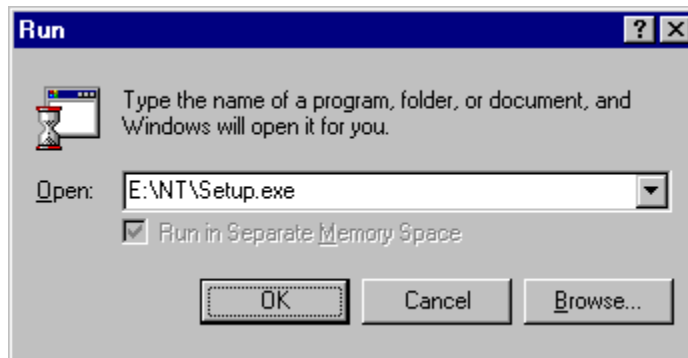


Figure 180. Installation Program's Name and Path

Notice that E in our case was the letter assigned to the CD-ROM drive. Click **OK** to run the installation routine. It will start to find all the JVMs installed on your system. Since we had already installed the JDK 1.1.6, the following window appeared:

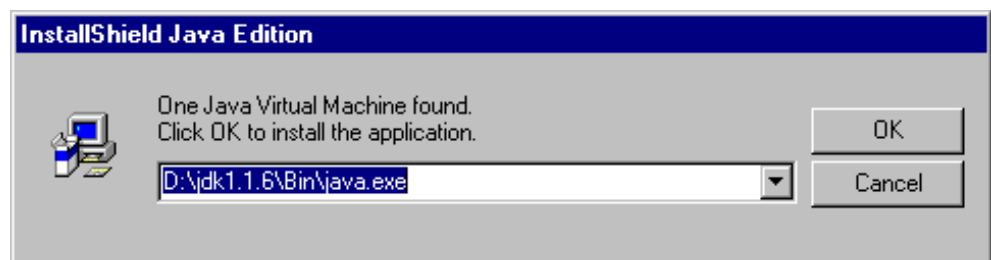


Figure 181. Selection of Java Virtual Machine Version

It's interesting to notice that if no JVMs are installed on your system, the installation routine reacts displaying the following panel:

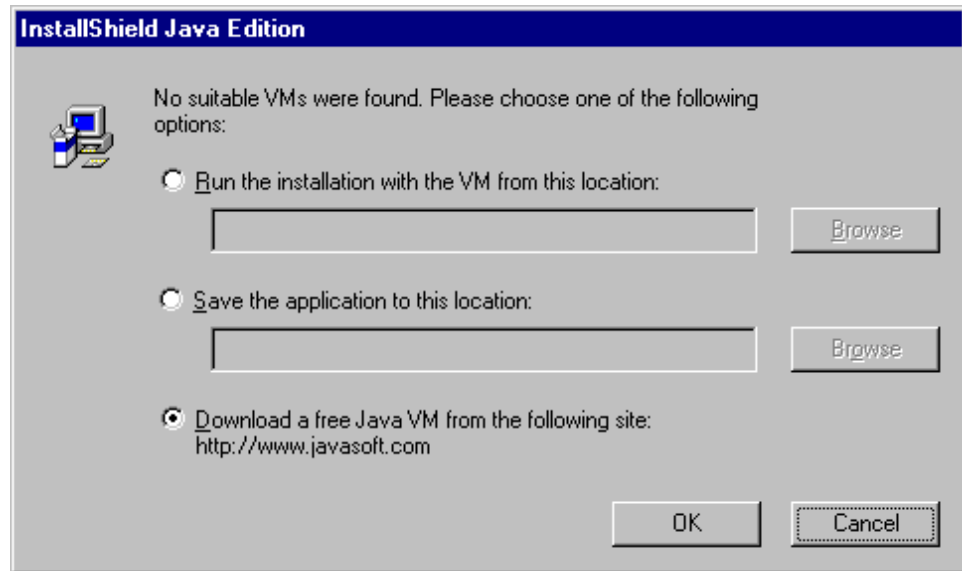


Figure 182. Choose Option If No JVM Is Installed

As you can see in the above figure, you have three options:

1. You can run the installation using a JVM located on a remote location.
If you select this option, the installation can be performed with a JVM from a remote machine.
2. You can save the Java InstallShield's setup.class on your disk.
If you choose this second option, the Java InstallShield's setup.class is copied onto a disk in your machine or in the network your machine belongs to, but the installation is not issued.
3. You can download the JVM from the JavaSoft Web site for free.
By selecting this option, your default Web browser automatically starts and points to the JavaSoft Web site. Also in this case the installation is not issued.

Since we had already installed one version of JVM, we clicked **OK** in the window displayed in Figure 181 on page 236 and we saw the Welcome window. After clicking the **Next** button, you are required to agree to all items of the software license. If you agree, check the box **Accept all terms of the license** and then click **Next**. You will get the a window displaying the IBM WebSphere Performance Pack Version 1.0 readme file. We suggest you take a look at that file as it contains interesting information about the product. Click again on **Next** and a dialog will be displayed where you can select the language to use, but only for the Load Balancing component, as shown in the following screen:



Figure 183. Load Balancing Component Language Selection

Notice that this screen appears even if you have not selected yet that you want to install the Load Balancing component.

After you click **Next**, you are prompted to enter the IBM WebSphere Performance Pack installation directory, which by default is named WebSphere, and we had the setup program create it on the D drive, as shown in the next window:

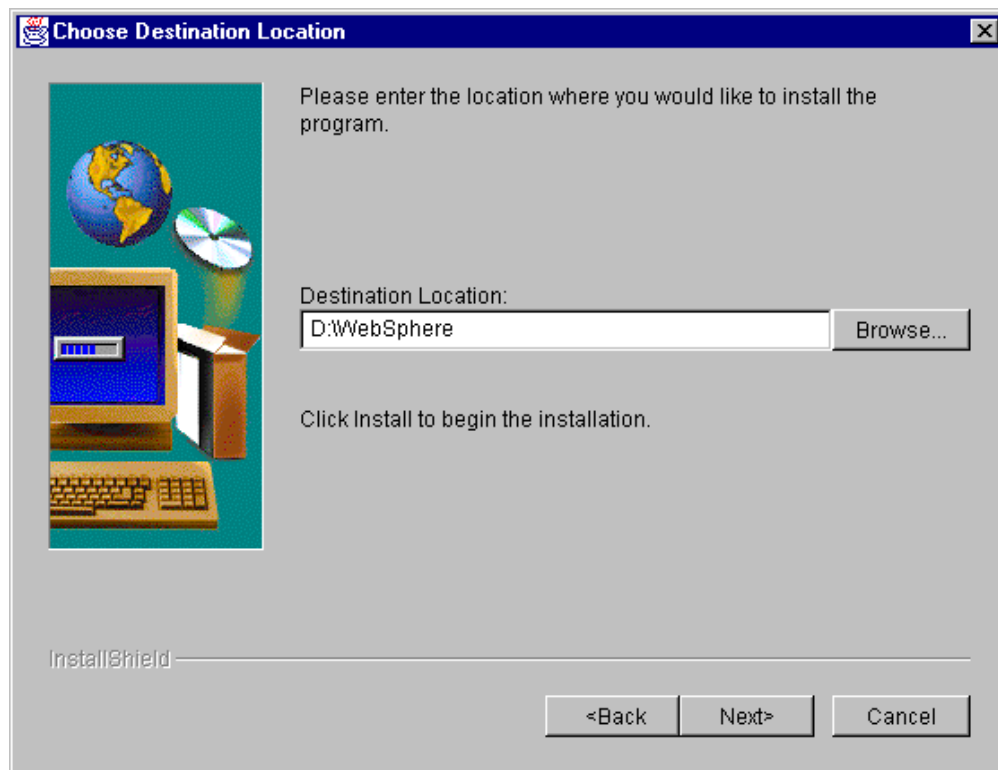


Figure 184. Choose Destination Location

Even if the above window invites you to click Install to continue with the installation, such a button does not exist. You should click on **Next** instead.

After that, select the IBM WebSphere Performance Pack component that you wish to install on your Windows NT platform.

Note that the installation program allows you to install on Windows NT only the following component combinations:

- Caching and Filtering component and/or File Sharing client component
- Load Balancing (eNetwork Dispatcher) component and/or File Sharing client component

In other words, you cannot install the Caching and Filtering component and the Load Balancing component on the same Windows NT machine. We selected **Load Balancing (eNetwork Dispatcher)**, and the Caching and Filtering check box appeared immediately, disabled.

Furthermore, note that you cannot select the check box File Sharing Server, since such a component is not available on Windows NT. This is the reason why the related check box appears disabled in the following window:



Figure 185. Select Component Window

After you click **Next**, you will get this window, which asks you if you want the installation program to replace other programs already installed on your system.

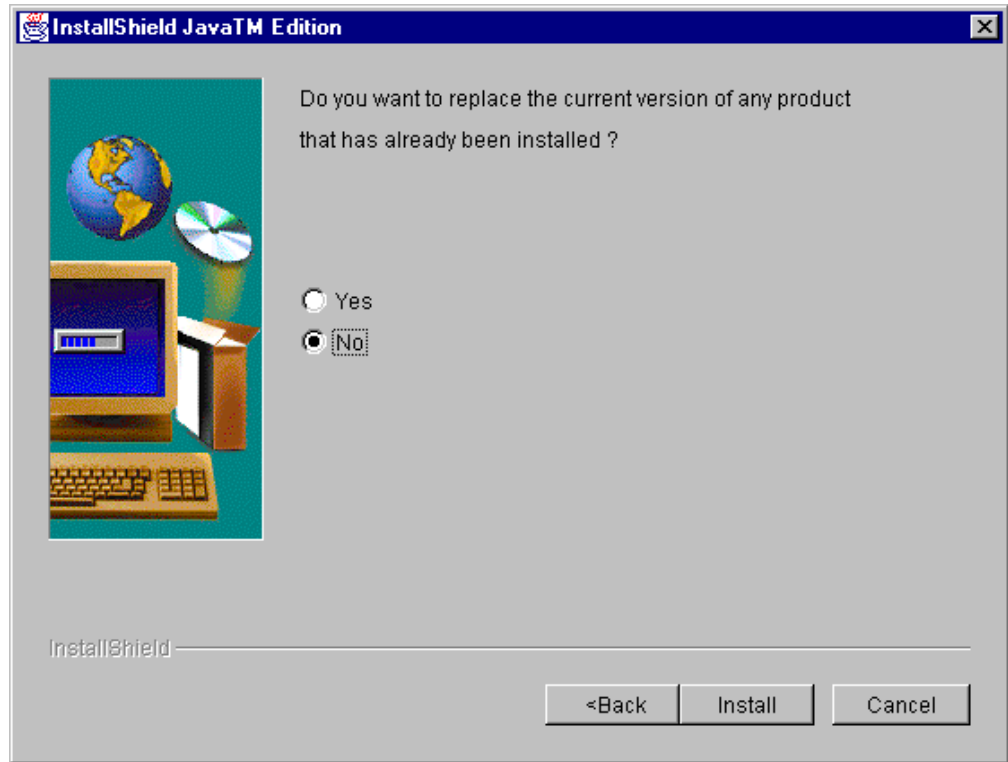


Figure 186. Choose to Replace Version

Since we had not installed any other IBM WebSphere Performance Pack components on that particular Windows NT machine, we chose **No**. (We also tried in another installation to select **Yes**, and we did not notice any differences.) Then we clicked **Next** and the installation routine started extracting the installation files onto the disk. At the end of this step the following window was shown:

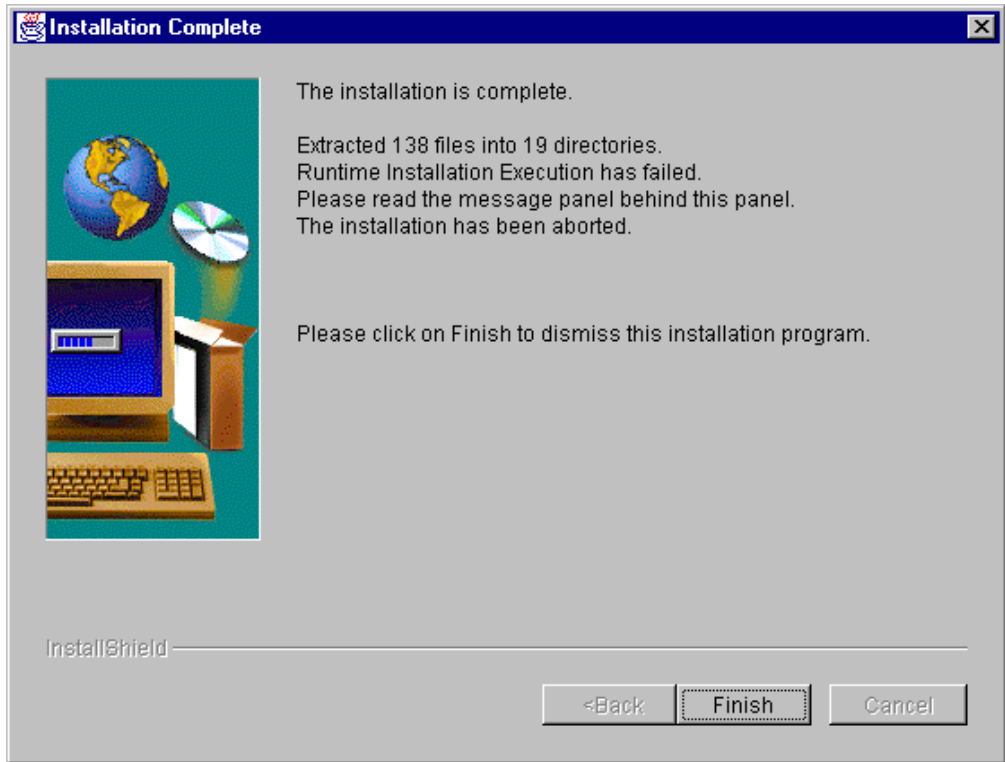


Figure 187. Installation Problem Window

Another window was brought up on our desktop:

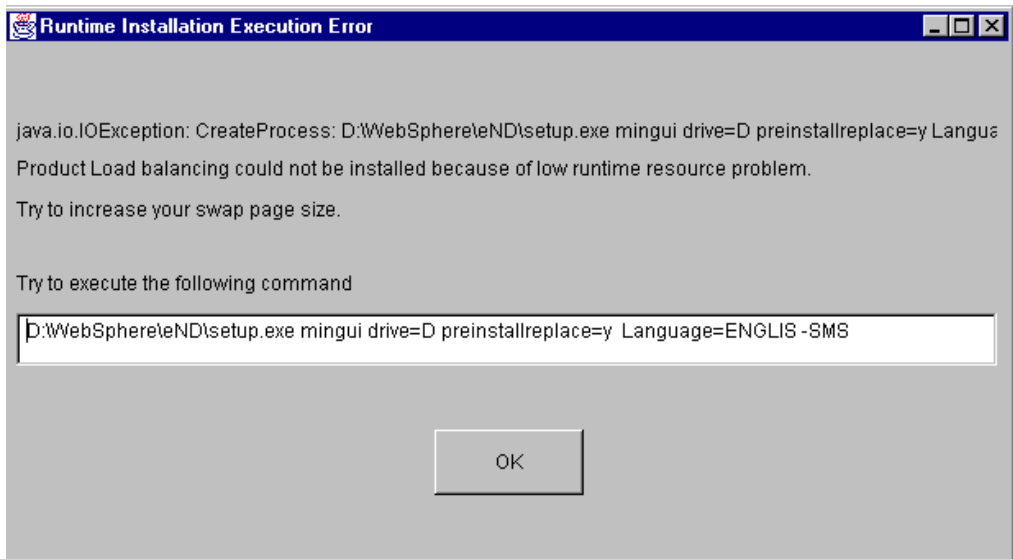


Figure 188. Runtime Installation Execution Error

This panel suggested we increase our swap page size. In our Windows NT machine, the total paging size for all disk volumes was minimum of 200 and maximum 400MB. (In detail, we had set two pagefile.sys files in our system, one in the C drive and one in the D drive, each having a size variable between 100 and 200MB.) We also tried another installation where the total paging size for all

disk volumes was minimum 250 and maximum 400MB. (In detail, the pagefile.sys file was 150-200MB sized in the C drive and 100-200MB sized in the D drive.) In this case the same problem also appeared.

The recommended command we copied and pasted in a MS-DOS window, as shown in the following figure:

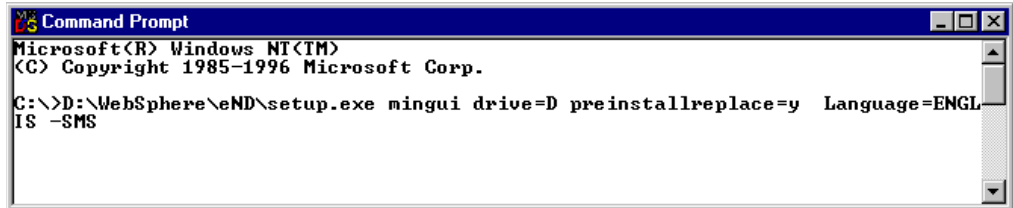


Figure 189. Installation Command

After entering the command shown in Figure 189 on page 243, the following window is displayed, by which you can select the type of installation you want to perform:

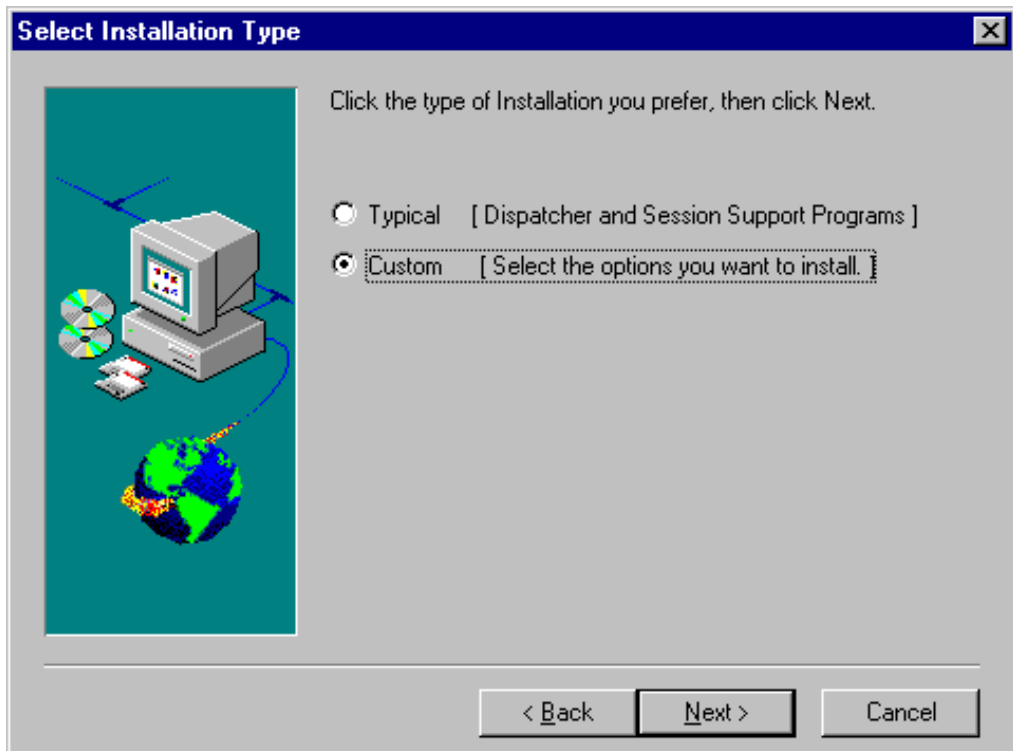


Figure 190. Selection of Installation Type

We checked off **Custom** since we wanted to have the ability to personalize the installation conforming to our needs, and then we clicked **Next**. Then we chose to install the two Load Balancing subcomponents (the Dispatcher and the Interactive Session Support) plus the documentation, as you can see in the following window:

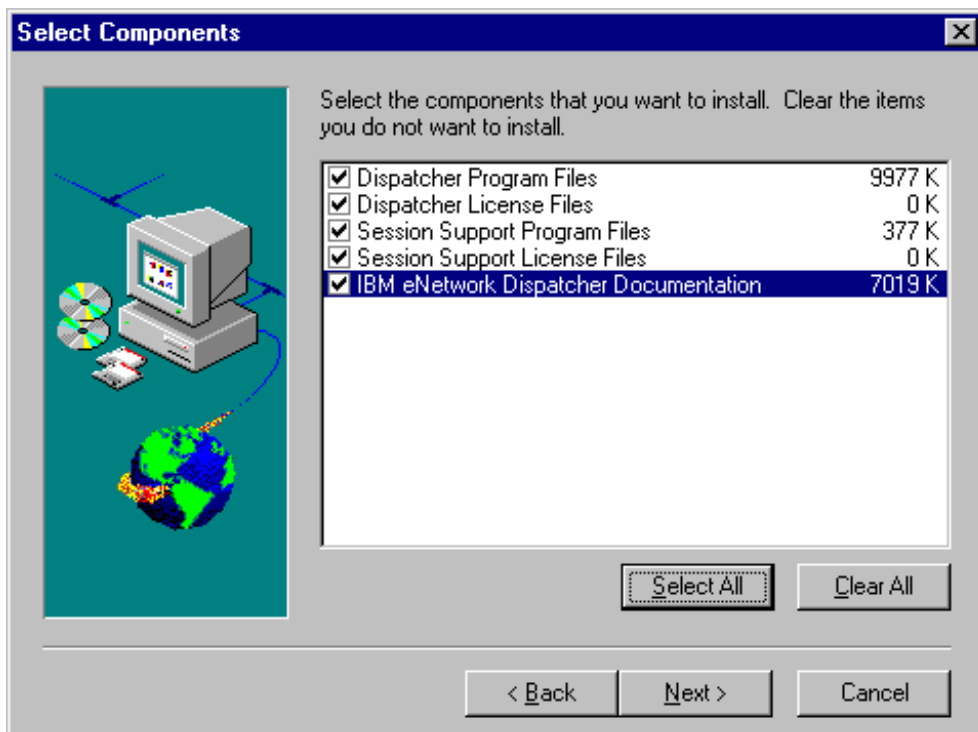


Figure 191. Selection of Components

To continue, select **Next**. The next dialog will be shown, for the selection of the Language files:



Figure 192. Select Language Files Window

As soon as you click **Next**, the above window disappears and the installation procedure goes on and completes without giving back any other signal. We did not receive any notification that the installation was completed and no panel was brought up to suggest we reboot our system, which usually happens after a complex installation.

You can realize that the installation has been completed if you check the services available on your computer in the Services dialog box of the Control Panel folder. You will see that both the IBM Network Dispatcher and IBM_ISS_Load_Balancing services have been created, even if they have not started yet, as shown in the following window:

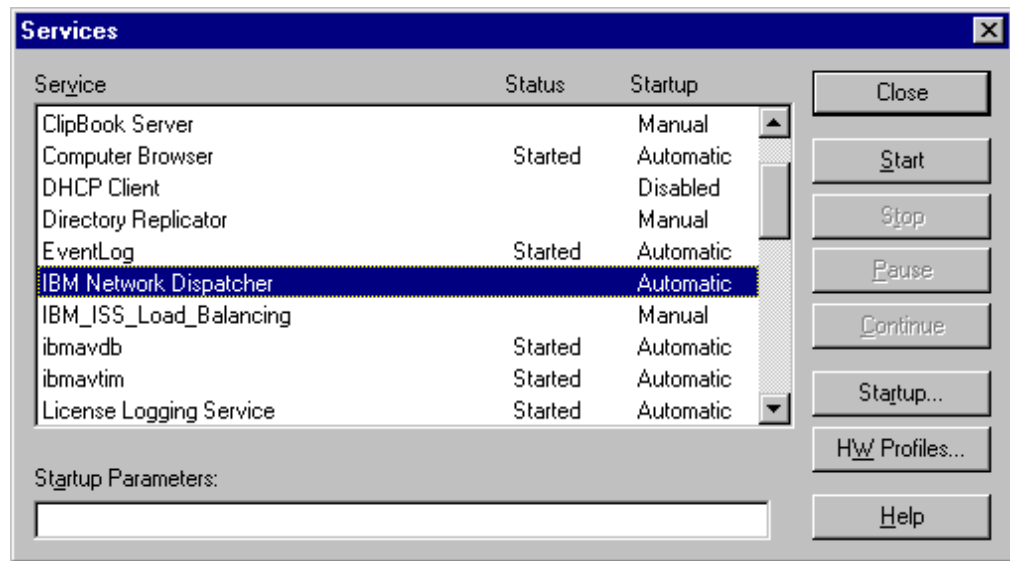


Figure 193. Available Services on Windows NT

Notice that the IBM Network Dispatcher service Startup mode is by default set to Automatic, while the IBM_ISS_Load_Balancing service has the Startup mode set to Manual. We tried to start that service by selecting the **Start** button, and the service Status changed to read *Started*.

Moreover, if you look at the Programs folder of the Start menu, you will find that the eNetwork Dispatcher icon (by which you start the GUI for the eNetwork Dispatcher configuration) is displayed. However, if you select it at this point, you get an error:

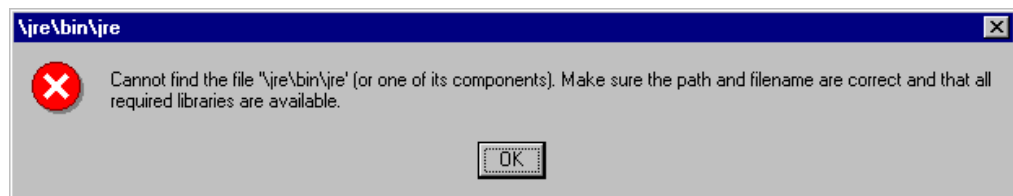


Figure 194. Execution Error

Reboot your system, and then you can find IBM Network Dispatcher service has really started.

Another thing to be emphasize is that in the directory \Temp of your boot logical disk, two files remain after this process. The first of them is a very large file, setup.class (over 26MB), and the second is getspace.exe. (You find this file also on the CD-ROM, in the directory where the setup.exe file is.) It is advisable to remove those files manually after rebooting the system.

4.6 Load Balancing Basic Scenario Scenario Using the Dispatcher

In this section we show you how to create a basic configuration for the Dispatcher. We built up a network environment with five workstations. On one of them we installed the Load Balancing component of IBM WebSphere Performance Pack, which worked to load balance three TCP/IP servers, namely Web servers and FTP servers. Another machine played the role of the Web client.

In our scenario, we installed our server machines with Lotus Domino Go Webserver Version 4.6.2.5.

4.6.1 Network Environment

A summary of the hardware, software and network configuration of the environment where we performed our test is reported in the following table:

Table 4. Basic Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM PC 365	wtr05193	9.24.104.239	Windows NT Server 4.0	Web Client
IBM RS/6000 43P	1) rs600023 2) clusterend	1) 9.24.104.128 2) 9.24.104.105	AIX 4.3.1	Load Balancing
IBM RS/6000 43P	rs600022	9.24.104.127	AIX 4.3.1	Web Server
IBM PC 365	wtr05219	9.24.104.44	Windows NT Server 4.0	Web Server
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	Web Server

On the three PCs installed with Windows NT Server 4.0, we also applied the Service Pack 3.

All the above machines were provided with a token-ring interface and connected to the same LAN.

Notice that:

1. The load balancing function was provided by the Load Balancing component of IBM WebSphere Performance Pack Version 1.0.
2. Netscape Navigator 4.06 was the Web browser running on the Web client machine.
3. The Web server function on the three clustered Web servers was provided by Lotus Domino Go Webserver Version 4.6.2.5.

The following figure offers a logical representation of the network environment where we performed this experience:

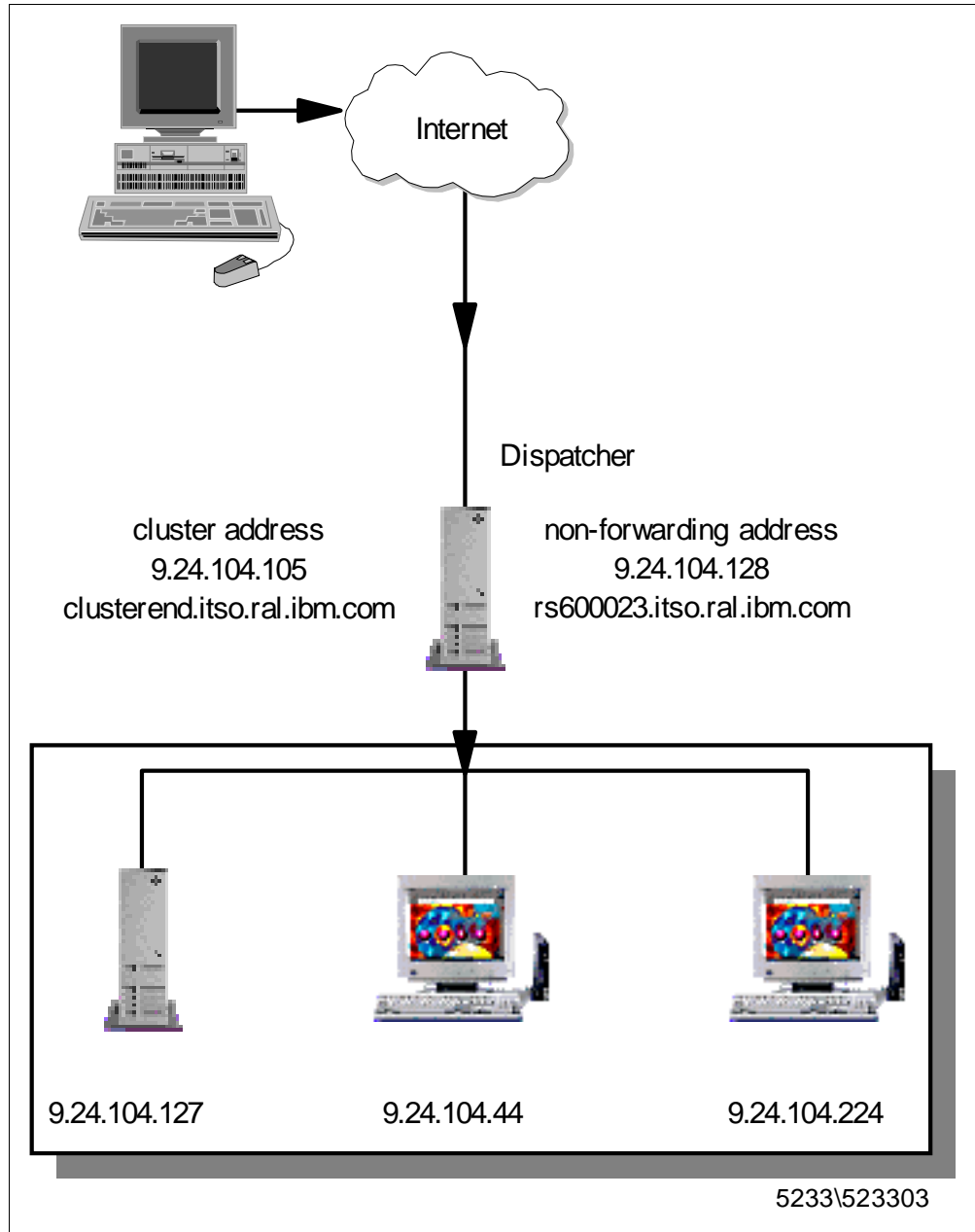


Figure 195. Graphical Representation of the Scenario Environment

4.6.2 Cluster Address and Non-Forwarding Address

Note that for the Load Balancing machine two addresses are needed:

1. The primary IP address for the Dispatcher machine is the IP address that is returned by the command:

```
host hostname
```

It is also called a *non-forwarding address*.

Through the non-forwarding address, you can connect to the machine for management purposes (using FTP or Telnet, for example). In our

configuration, the non-forwarding address was 9.24.104.128. In fact, by issuing the command `host rs600023` the output produced was:

```
rs600023.itso.ral.ibm.com is 9.24.104.128
```

2. The *cluster address* is the IP address that will be used by clients to access the entire site. The dispatcher's work will be dedicated to make load balancing of this address. An host name can be associated to the cluster address. In our configuration, the cluster address of the load balancing machine was 9.24.104.105 and the associated host name was `clusterend.itso.ral.ibm.com`. Notice that you need to define a cluster address for each cluster you are going to define in your environment.

Cluster Address and Network Administration

The cluster address is the unique IP address by which client requests access your cluster. It is not a virtual address valid only locally. No other machine in the network should be given that address. For this reason, you must contact the network administrator before assigning a cluster address and the associated host name to the Dispatcher.

All of our workstations were located in the same LAN, and each of them was provided with only one token-ring network interface card. We also ensured that the workstations could ping each other, and that the three Web servers hosted the same Web content. Notice that this configuration involved only the Load Balancing component of IBM WebSphere Performance Pack. We made sure that the three Web servers hosted the same contents by simply duplicating the same Web pages on the three machines. In this case we did not use the File Sharing component of IBM WebSphere Performance Pack.

The configuration process can be divided into two logical phases: the setup of the Dispatcher machine and the setup of the TCP/IP server machines. Besides the issues typically related to the Dispatcher functionalities, you have to face also some TCP/IP configurations steps. If you want to have detailed explanations about TCP/IP, you can refer to the IBM redbook *TCP/IP Tutorial and Technical Overview*, GG24-3376.

4.6.3 Dispatcher Configuration

In the following, we refer to the configuration of the Dispatcher on the AIX platform. The configuration process on Windows NT is very similar; if something differs, we explicitly mention it.

4.6.3.1 Configuration Methods

Three different methods can be adopted to configure the Dispatcher:

1. Command line

You can enter the configuration commands from a command prompt.

2. Scripts

You can write the commands into a configuration file script that acts as a batch file. Running that script, the commands can be executed all together.

3. Graphical user interface

Version 2.0 of the Load Balancing (eNetwork Dispatcher) component introduces this new feature, which we found very easy-to-use: the user on the Load Balancing machine is able to perform all the configuration steps strictly related to the dispatcher function just using the graphical user interface (GUI). In fact, by using the GUI, you can issue the `ndcontrol` command with the parameters and flags that are used to configure the Dispatcher machine, as we are going to explain. However, you cannot use the GUI for TCP/IP-related commands such as `ifconfig`, so this kind of command can be entered only from the command line.

In our configuration process we made wide use of the GUI and we noticed that it is really possible to use both the GUI and command line to issue the Dispatcher configuration unless some special needs require you to interact with the TCP/IP configuration. In the following, for each configuration step, we also show the correspondent command entered on a command prompt.

In order to perform the configuration steps listed below, you must be the root user on AIX or, if the Dispatcher is installed on Windows NT, a system administrator.

4.6.3.2 Starting the Server Component

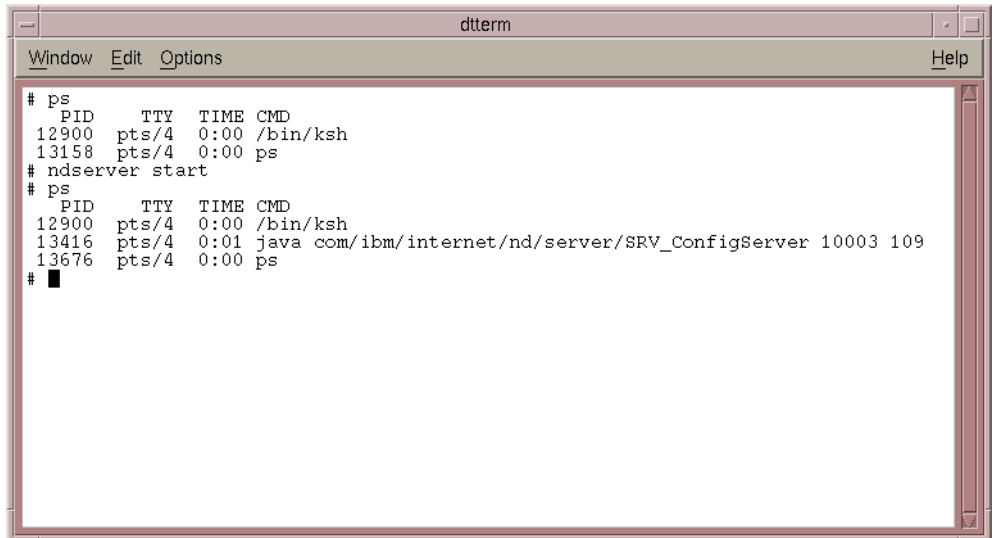
First of all, you need to start the server component of the Dispatcher. If on AIX, from a command line, enter:

```
ndserver start
```

Note that on Windows NT the server component is a Windows NT service that starts automatically by default. The name of this service is IBM Network Dispatcher. If you want to stop or restart the IBM Network Dispatcher service, you should open the Services window of the Windows NT Control Panel (see Figure 193 on page 245), highlight **IBM Network Dispatcher** and then click **Stop** or **Start** respectively.

If you prefer that the IBM Network Dispatcher service does not start by default at each system reboot, select **IBM Network Dispatcher**, click on **Startup...**, and then set the Startup Type to **Manual**.

On AIX, if you enter the `ps` command after launching the command `ndserver start` the server component is implemented as a Java class, running through the JVM java executable file:



```
# ps
  PID   TTY   TIME CMD
 12900 pts/4 0:00 /bin/ksh
 13158 pts/4 0:00 ps
# ndserver start
# ps
  PID   TTY   TIME CMD
 12900 pts/4 0:00 /bin/ksh
 13416 pts/4 0:01 java com/ibm/internet/nd/server/SRV_ConfigServer 10003 109
 13676 pts/4 0:00 ps
# █
```

Figure 196. Server Component of the Dispatcher

4.6.3.3 Starting the GUI

Now you can start the GUI. To do so, enter the command `ndadmin`.

If your platform is Windows NT, you can also click on **Start**, select **Programs** and then click on **Network Dispatcher**. The Network Dispatcher item in the Programs menu was created during the installation.

After a while, the GUI will be displayed:

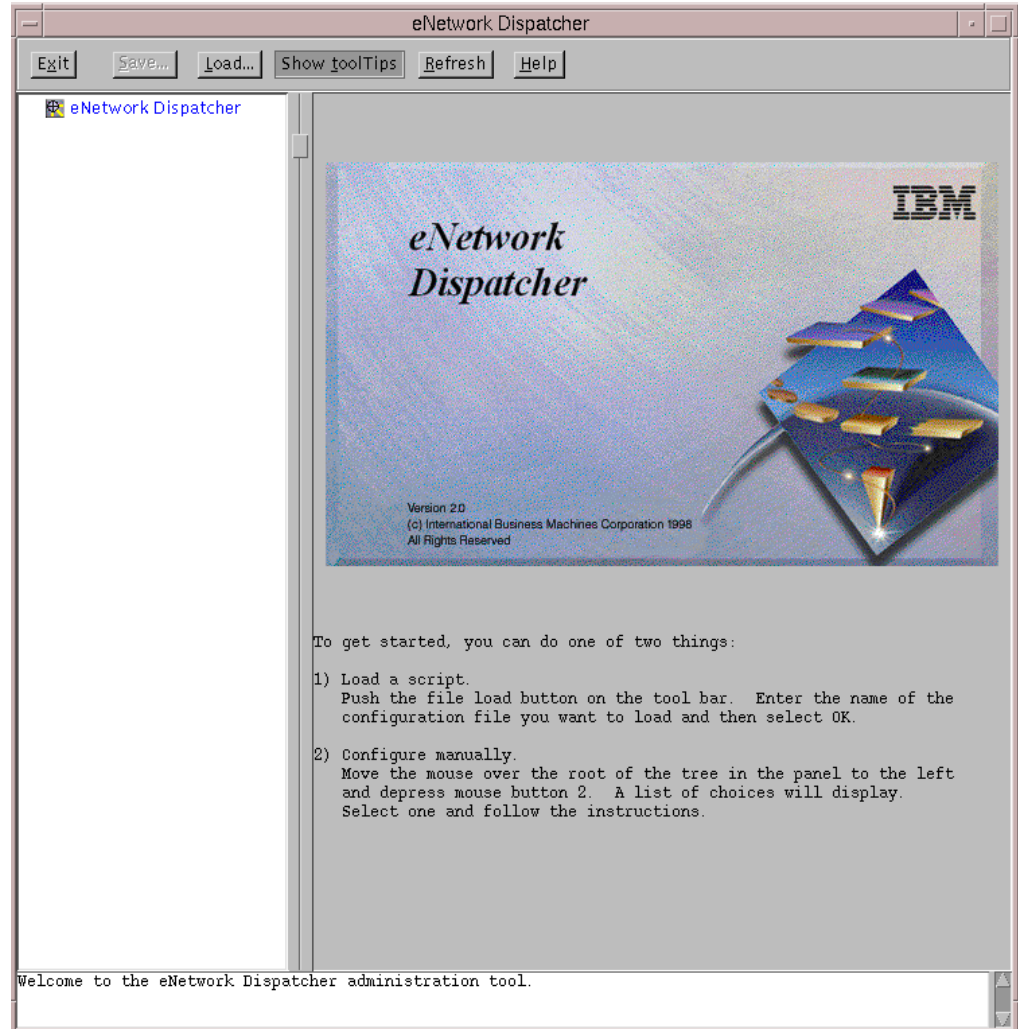


Figure 197. Graphical User Interface of the Dispatcher Configuration

The GUI works by using Java classes. We want to mention that while configuring the Dispatcher on AIX, the background command prompt window from which you launched the `ndadmin` command may register a `java.lang.NullPointerException`, as shown in the following screen:

```
java.lang.NullPointerException
    at java.awt.LightweightDispatcher.retargetMouseEvent(Compiled Code)
    at java.awt.LightweightDispatcher.processMouseEvent(Compiled Code)
    at java.awt.LightweightDispatcher.dispatchEvent(Compiled Code)
    at java.awt.Container.dispatchEventImpl(Compiled Code)
    at java.awt.Window.dispatchEventImpl(Compiled Code)
    at java.awt.Component.dispatchEvent(Compiled Code)
    at java.awt.EventDispatchThread.run(Compiled Code)
```

If you should notice a similar event, you can safely continue using the GUI to configure the Dispatcher. In fact, the above is not a real problem and does not affect the Dispatcher configuration.

4.6.3.4 Starting the Executor

The first thing you have to do now is to start the Executor component of the Dispatcher. The Executor is the Dispatcher component that routes requests to the TCP and UDP servers. It also monitors the number of new, active and finished connections and performs garbage collection of complete or reset connections. The Executor supplies information about the new and active connections to the Manager component.

To start the Executor, right-click with your mouse in the left white area of the above window, and in the pop-up window that appears select **Start Executor**, as shown in the next figure:

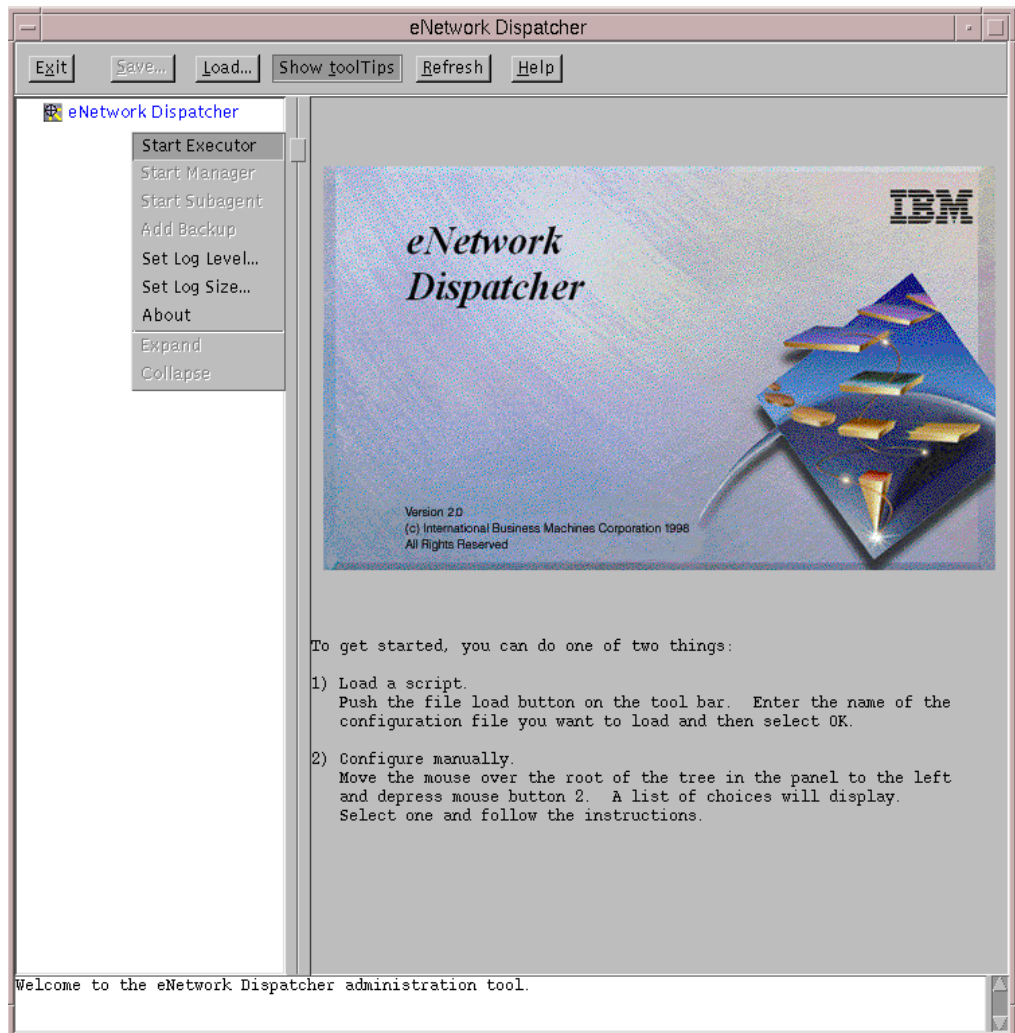


Figure 198. Start Executor

You will get a window similar to the following:

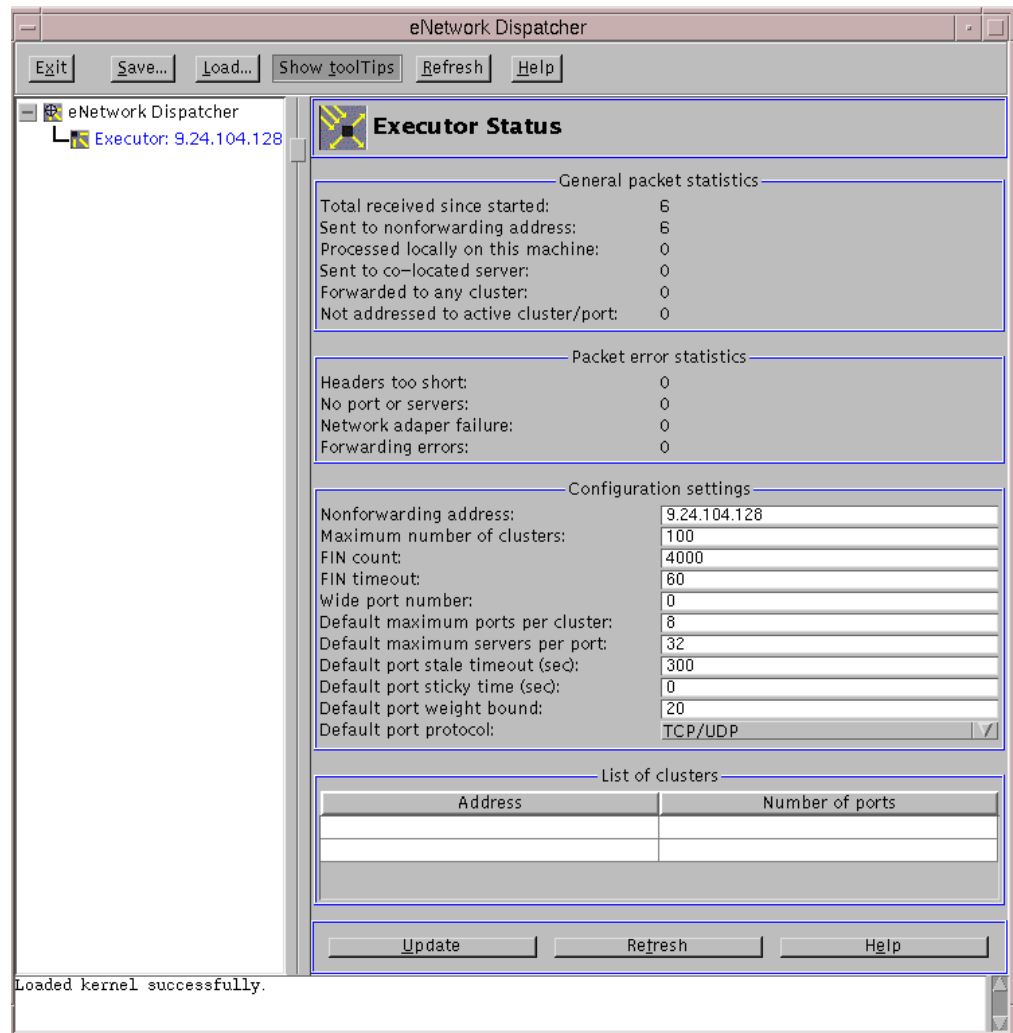


Figure 199. Executor Status Window

Alternatively, you can start the Executor component from a command line by entering the command:

```
ndcontrol executor start
```

4.6.3.5 Configuring the Executor

At this point, you might want to re-define the non-forwarding address or the host name of the Load Balancing machine. In our configuration the non-forwarding address appeared by default in the Configuration settings section of the Executor Status window as 9.24.104.128 (which was the IP address assigned to the network interface card of our machine), and we did not need to change that value (see also Table 4 on page 246). However, there are cases where machines have more than one network interface card and administrators may want to use one for cluster traffic and the other for administrative purposes. In those cases, the non-forwarding address displayed by default might not be the right one, and then it becomes necessary to modify it by entering the proper value in the Configuration settings section of the Executor Status window.

Using the Command Line

To set the non-forwarding address from the command line, you can enter the following command:

```
ndcontrol executor set nfa IP_address
```

Put the non-forwarding address in place of *IP_address*. For example, on our platform, the full command should have been:

```
ndcontrol executor set nfa 9.24.104.128
```

However, as we said, we did not need to perform this step, since the exact non-forwarding address by default appeared in the Executor Status window.

More About the Non-Forwarding Address

The non-forwarding address has a special meaning inside the kernel code. Any packets that have the non-forwarding address as the destination address are immediately given back to the local operating system for processing. No Dispatcher logic is used to determine where the packet should go. This check is done immediately after receiving the packet.

Notice that no error checking is done to see that the non-forwarding address is an existing IP address on that machine, so you could even set the non-forwarding address to an IP address that has not been defined on that machine, and your command would be accepted, even if you could not administer the Dispatcher machine using such an address.

Another use of the NFA is for co-located servers. If the Dispatcher logic determines that the destination server has the non-forwarding address as its address, the packet is given back to the underlying operating system, not routed out the network interface. This implies that in order for co-location to work, the server on the local machine must be added with the non-forwarding address.

The Configuration settings section lists a set of parameters for the Executor that you can change. Some of them have immediate meaning, such as the Maximum number of clusters (default is 100), Default maximum ports per cluster (default is 8) and the Default maximum servers per port (default is 32). Others are more difficult to understand, such as the FIN count and the FIN timeout. FIN is a control bit occupying one sequence number, which indicates that the sender will send no more data. A client sends a FIN after it has sent all its packets, so that the server will know that the transaction is finished. When the Dispatcher receives a FIN, it marks the transaction from active state to FIN state, and at this point the Dispatcher garbage collector can clear the memory reserved for those connections.

For a stable running configuration, you can safely leave the parameter values that appear by default in the Configuration settings section of the Executor Status window unchanged. For detailed information about the parameters, you can consult the eNetwork Dispatcher help, which can be accessed by selecting the **Help** button on the top menu bar, or see the document *eNetwork Dispatcher for*

Solaris, Windows NT and AIX User's Guide, GC31-8496, shipped with the CD-ROM of IBM WebSphere Performance Pack.

4.6.3.6 Aliasing the Network Interface to the Cluster Address

Now you need to create an alias on the Dispatcher machine's network interface card to the cluster address. To do this, you cannot use the GUI. If your platform is AIX, you should use instead the TCP/IP command `ifconfig`. The full command syntax on AIX is as follows:

```
ifconfig interface_name alias cluster_address netmask netmask
```

where:

- *interface_name* is the name assigned to the network interface.
On our platform, the network interface name was `tr0` since our machine was equipped with a token-ring network adapter.
- *cluster_address* is the cluster address assigned to the Dispatcher machine.
In our case we entered `9.24.104.105`, according to Table 4 on page 246.
- *netmask* is the netmask value you are using in the TCP/IP configuration of the Dispatcher machine. It can be specified either in dotted-decimal or hexadecimal form.

In our case we could have entered either `255.255.255.0` or `0xffffffff00`.

If your platform is Windows NT, you should instead use the command `ndconfig`. Actually, this command does not belong to the set of commands related to the TCP/IP protocol on the Windows NT operating system. However, while installing the Load Balancing component of IBM WebSphere Performance Pack, the installation process locates the executable `ndconfig.exe` in the directory `D:\Program Files\END\Dispatcher\BIN`, and the same installation process updates the value of the Path system environment variable, adding the complete path to `ndconfig.exe`. The syntax of the `ndconfig` command is the same as the AIX command `ifconfig`.

It is recommended you know exactly the value of the netmask parameter as it was set on your Dispatcher machine. If you fail or omit it, an unwanted route to the IP address might be created in the routing table.

In order to know what the value of the netmask parameter is, on AIX you can enter the command `ifconfig` followed by the name of the network interface. If the platform is Windows NT, `ifconfig` should be replaced by `ndconfig`.

For example, on the Dispatcher machine that we installed on AIX, we entered the command:

```
ifconfig tr0
```

The output we received is displayed in the following screen:

```

dtterm
Window Edit Options Help
# ifconfig tr0
tr0: flags=e0a0843<UP,BROADCAST,RUNNING,SIMPLEX,ALLCAST,MULTICAST,GROUPRT,64BIT>
      inet 9.24.104.128 netmask 0xfffff00 broadcast 9.24.104.255
# █

```

Figure 200. *ifconfig* Command Entered to Discover the netmask Value

So, on our platform, the full command to create the alias on the network interface tr0 to the cluster address 9.24.104.105 was:

```
ifconfig tr0 alias 9.24.104.105 netmask 255.255.255.0
```

To ensure that the alias has been created correctly, enter the command:

```
netstat -ni
```

You should get two IP addresses for your network interface. The following figure shows the output we got from the `netstat -ni` command before and after we created an alias on the Dispatcher network interface using the `ifconfig` command:

```

dtterm
Window Edit Options Help
# ifconfig tr0
tr0: flags=e0a0843<UP,BROADCAST,RUNNING,SIMPLEX,ALLCAST,MULTICAST,GROUPRT,64BIT>
      inet 9.24.104.128 netmask 0xfffff00 broadcast 9.24.104.255
#
#
# netstat -ni
Name Mtu Network Address Ipkts Ierrs Opkts Oerrs Coll
lo0 16896 link#1 2507 0 2525 0 0
lo0 16896 127 127.0.0.1 2507 0 2525 0 0
lo0 16896 ::1 2507 0 2525 0 0
tr0 1492 link#2 8.0.5a.ce.6d.7f 50078 0 593 0 0
tr0 1492 9.24.104 9.24.104.128 50078 0 593 0 0
#
# ifconfig tr0 alias 9.24.104.105 netmask 255.255.255.0
#
# netstat -ni
Name Mtu Network Address Ipkts Ierrs Opkts Oerrs Coll
lo0 16896 link#1 2507 0 2525 0 0
lo0 16896 127 127.0.0.1 2507 0 2525 0 0
lo0 16896 ::1 2507 0 2525 0 0
tr0 1492 link#2 8.0.5a.ce.6d.7f 50450 0 594 0 0
tr0 1492 9.24.104 9.24.104.128 50450 0 594 0 0
tr0 1492 9.24.104 9.24.104.105 50450 0 594 0 0
# █

```

Figure 201. *How to Verify That the Alias Has Been Created Correctly*

Cluster Addresses and Network Interfaces

Note that it is possible to configure the Dispatcher to manage more than one cluster, whether your Dispatcher machine is shipped with one network interface card or more than one. Let's see some examples.

- If your Dispatcher machine has only one network interface adapter, you can configure as many aliases as the number of clusters you want to define, on the same network interface card.
- If your Dispatcher machine is shipped with two network interface cards (for example, one Ethernet card, en0, and one token-ring card, tr0), and you want to manage two clusters, one on en0 and the second on tr0, then each cluster address must be aliased on the corresponding network interface card.
- One more example is if your Dispatcher machine is shipped with two network interface cards (one Ethernet card, en0, and one token-ring card, tr0), and you want to manage five clusters, say three on en0 and two on tr0. The first three cluster addresses should be aliased on tr0, and the other two cluster addresses on en0.

No limits are known to the number of cluster addresses that can be aliased on the same network interface card.

4.6.3.7 Aliases Configuration: Using Scripts

For the Dispatcher to route packets, each cluster address must be aliased to a network interface card.

Each time the Executor is started, you will have to again configure all the aliases for the network interfaces of the Dispatcher machine. To avoid manually configuring the network interface every time, you can use one or more script files, which are automatically launched by the Dispatcher as different conditions occur, or, as the Dispatcher changes its own *state*. For example, when using the high availability feature (see 4.11, "Dispatcher High Availability" on page 313) the Dispatcher begins to monitor the conditions of the currently active Dispatcher, without routing any packets; it is said that the Dispatcher goes into *standby* state.

Another example is after the Dispatcher finishes monitoring the health of the currently active machine. Then it changes its state, goes into the *active* state and begins routing packets.

Every time the Dispatcher changes its own state, a correspondent script is launched. For example, when the Dispatcher goes into the *idle* state, the related script that is launched is *goldle*. The Dispatcher goes into the *idle* state when it begins routing packets in a stand-alone configuration.

Sample versions of script files that accomplish these functions are located in the eND sample directory. On our systems, the eND sample directory was `/usr/lpp/eND/dispatcher/bin/samples/en_US` on AIX, and `D:\Program Files\eND\dispatcher\BIN\samples\en_US` on Windows NT.

As suggested in the document *eNetwork Dispatcher for Solaris, Windows NT and AIX User's Guide*, GC31-8496, we used the *goldle.sample* file that is particularly

indicated in case you are not implementing the high availability feature for the Dispatcher configuration. (For further details on the high availability, see 4.12, “Dispatcher High Availability Scenario” on page 314.)

We made a backup copy of that file, then we modified the original one by inserting into it the correct command line to create an alias on the network interface to the cluster address, as shown in the following figure:

```
#!/bin/ksh
#
#goIdle script
#
#This script is executed when a Network Dispatcher goes into
#the 'idle' state and begins routing packets. This occurs when
#the high availability features have not been added, as in a
#standalone configuration. It also occurs in a high availability
#configuration before the high availability features have been
#added or after they have been removed. goIdle should not be
#used in conjunction with high availability, it's intended
#purposes were for a standalone environment.
#
# This script must be placed in the /usr/lpp/eND/dispatcher/bin
# directory in order to be executed
#
ifconfig tr0 alias 9.24.104.105 netmask 255.255.255.0
```

Figure 202. Script File goldle.sample on AIX

In order for the above script to run, you are required to name the file as goldle (without any extensions) and place it in the Dispatcher bin directory, which by default is the /usr/lpp/eND/dispatcher/bin directory on AIX and D:\Program Files\eND\dispatcher\BIN on Windows NT.

4.6.3.8 Adding a Cluster

Now you need to define a cluster. From the GUI, right-click on the left panel (on the **Executor** item or below it), and in the pop-up window that appears select **Add Cluster...**, as shown in the next figure:

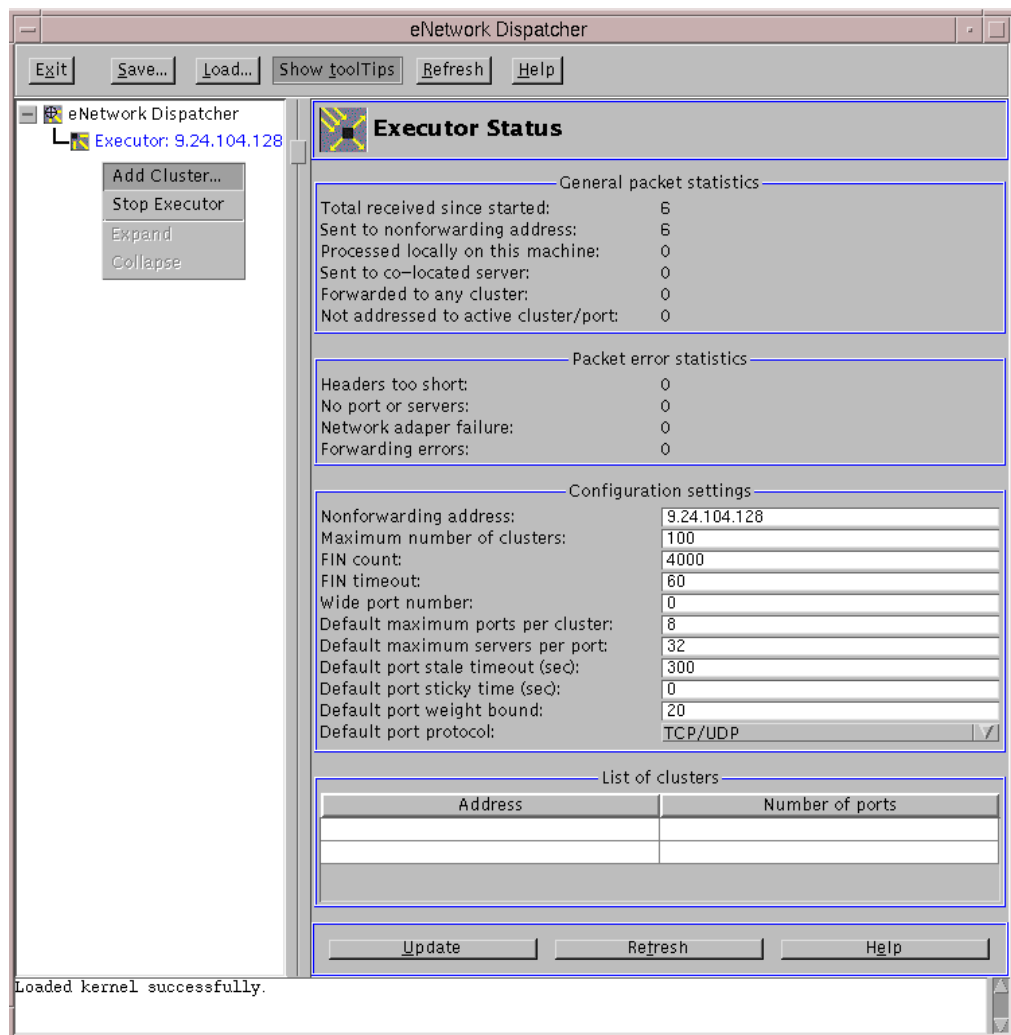


Figure 203. Adding a Cluster

Then type the cluster address and select **OK** in the panel that is brought up:

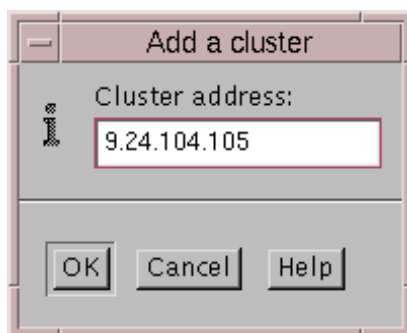


Figure 204. Enter the Cluster Address

Using the Command Line

If instead of the GUI you selected to use the command line, then you should enter:

```
ndcontrol executor set nfa cluster_address
```

In our case the command would have been:

```
ndcontrol executor set nfa 9.24.104.105
```

You will get the following Cluster status window, which confirms that a cluster has been added to the configuration:

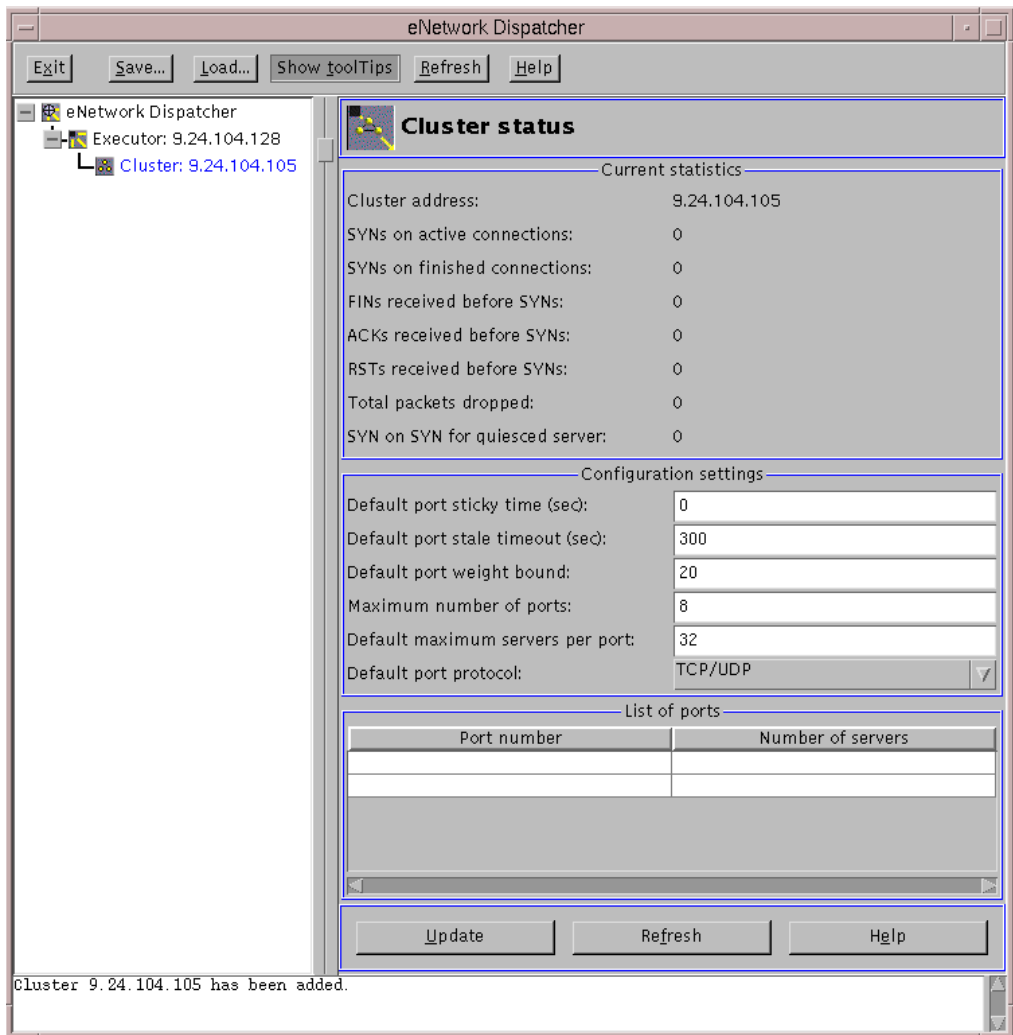


Figure 205. Cluster Added

4.6.3.9 Adding Ports

The Dispatcher machine can use several protocols to communicate with the TCP/IP servers. You need to define as many ports as the protocols you want the

Dispatcher machine to communicate through. Below is a table that lists the available protocols and assigned ports:

Table 5. Protocol and Port Used by the Dispatcher

Protocol	Port
FTP	20, 21
Telnet	23
SMTP	25
HTTP	80
POP3	110
NNTP	119
SSL	443

The port values shown in Table 5 on page 261 reflect the assigned port numbers provided by the Networking Working Group in the Request For Comments (RFC) 1700, as you can see at <http://info.internet.isi.edu/in-notes/rfc/files/rfc1700.txt>.

Note that if you want to use the FTP protocol, you should add the port 20 (also known as the FTP *control port*), which is the port through which an FTP server is listening for the commands, and the port 21 (also known as FTP *data port*), which is the port through which the data is transmitted during an FTP connection.

To add ports, from the GUI right-click on the left panel (on the **Cluster** item or below it), and in the pop-up window that appears select **Add Port...**, as shown in the next figure:

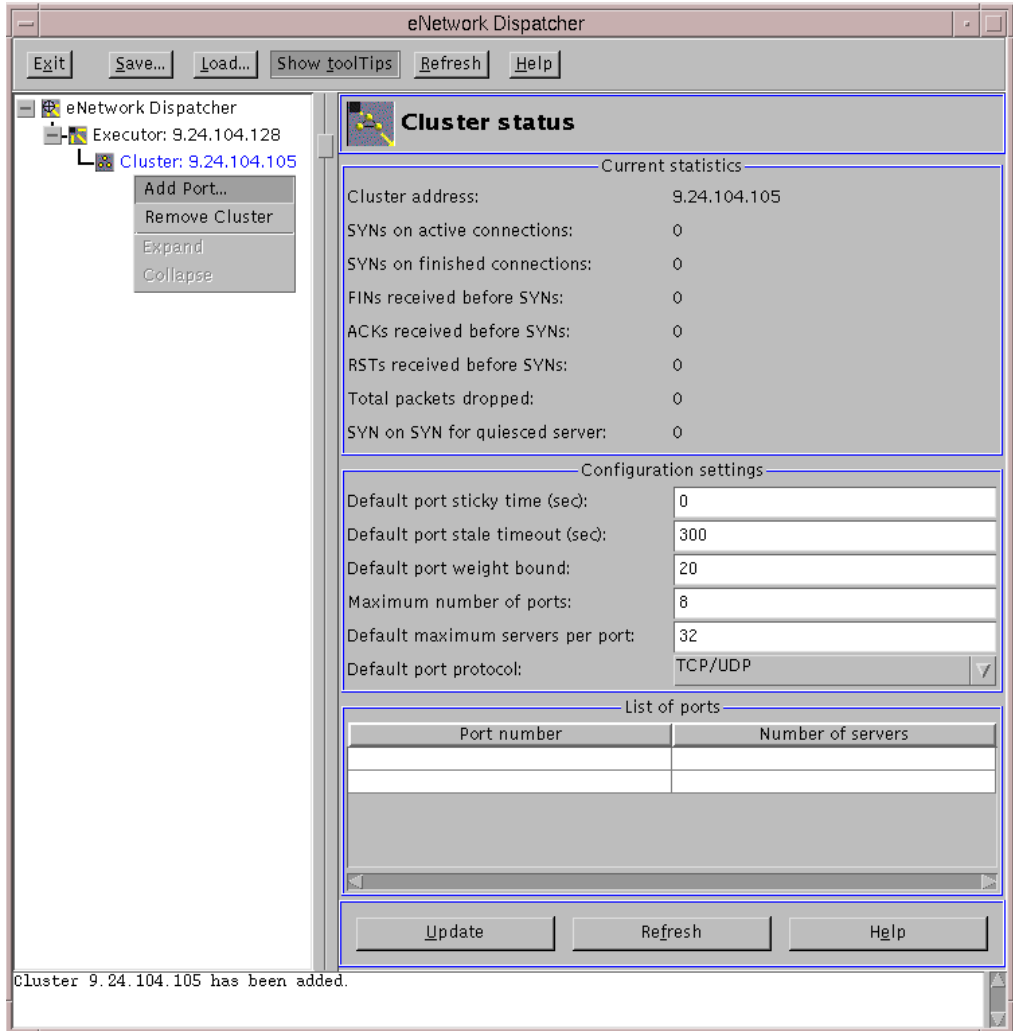


Figure 206. Adding a Port

Then type the port number you want and click **OK** in the panel that is brought up:

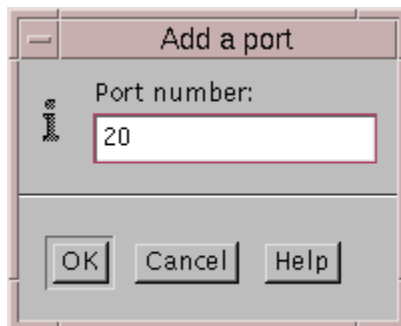


Figure 207. Enter the Port Number

First of all, we entered 20, as shown in Figure 207 on page 262. Following the same process we also added ports 21 and 80 to the same cluster using the GUI. We also added port 443, assigned to the SSL protocol, but in this case we wanted to experiment with the command line, rather than using the GUI again, and so we entered:

```
ndcontrol port add 9.24.104.105:443
```

Coming back to the GUI, and clicking the **Refresh** button in the top menu bar, we got the updated configuration with the four ports added.

Multiple Ports Using the Command Line

Note that using the command line you are able to add multiple ports entering just one command. In our configuration, in order to add all four ports to the cluster, we could have entered:

```
ndcontrol port add 9.24.104.105:20+21+80+443
```

4.6.3.10 Adding Servers

At this point of the configuration process you need to define the TCP/IP server machines that you want in the cluster. To do so, we right-clicked on the **Port 20** item and chose **Add Server...**, as shown in the following figure:

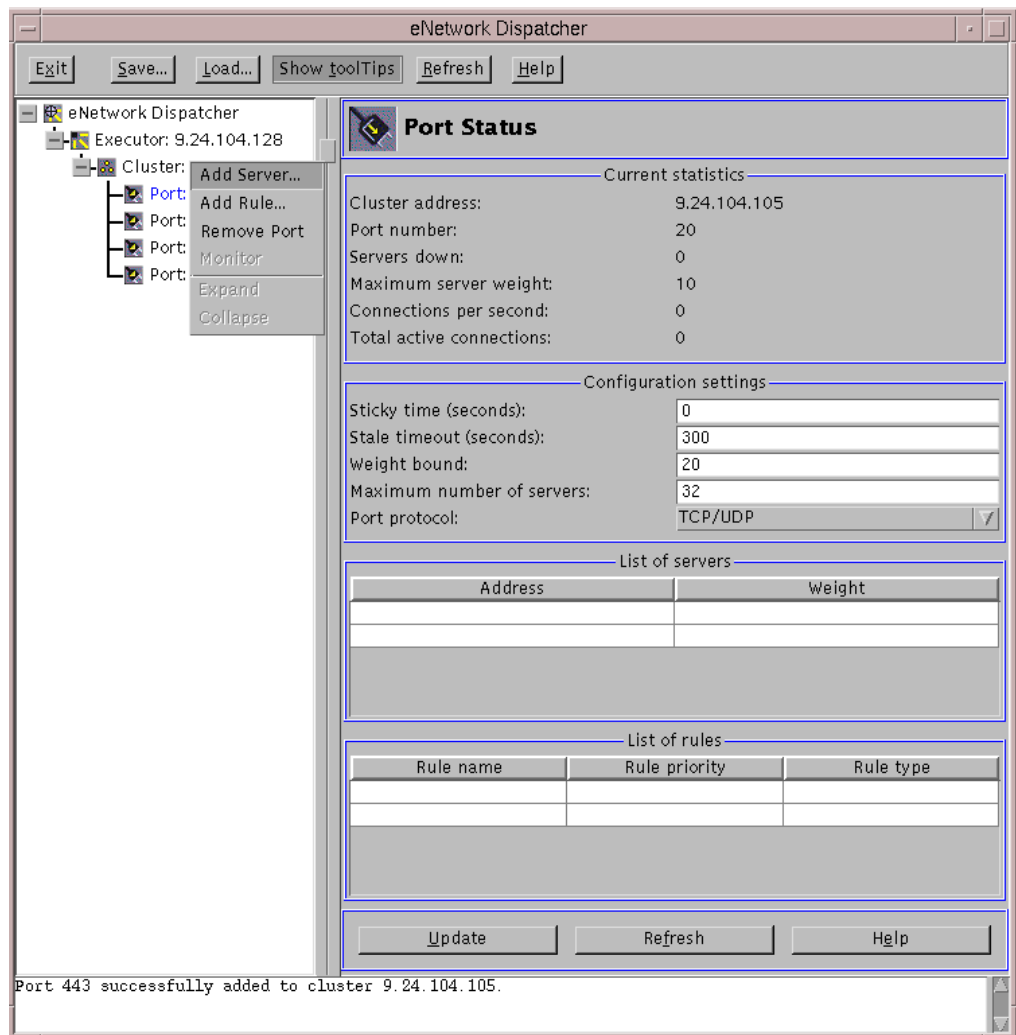


Figure 208. Adding a Server

Then type the server name and click **OK** in the panel that appears:

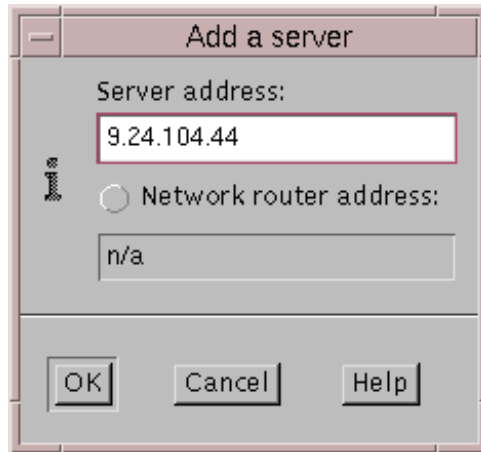


Figure 209. Enter Server Address

You can optionally click the **Network router address** radio button and fill in the network router address if you are defining a Dispatcher for a wide area network (WAN) configuration.

After clicking on **OK**, you will get back to the GUI. Expanding the **Port 20** item, we ensured that the server having IP address 9.24.104.44 had been added to the port 20 of the cluster having IP address 9.24.104.105, as shown in the following window:

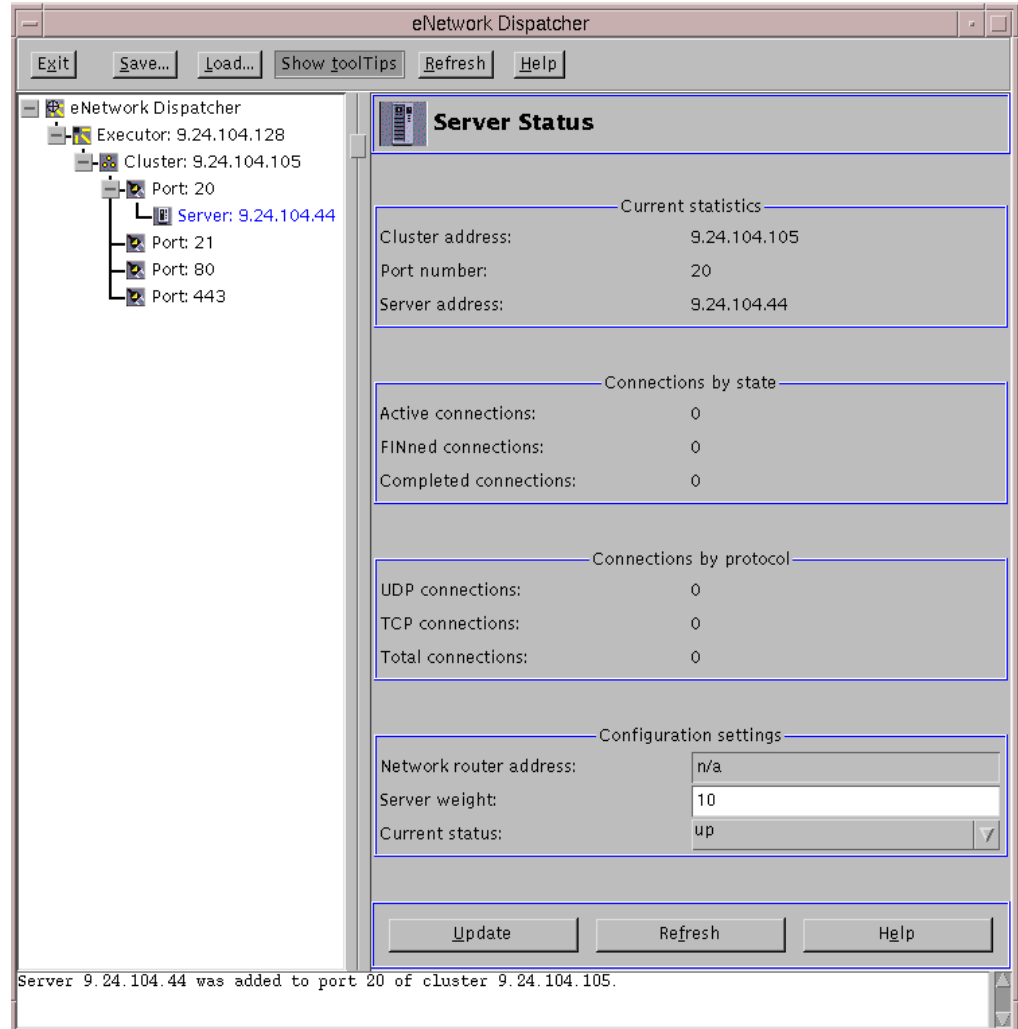


Figure 210. Verifying That the Server Has Been Added

We added through the GUI all the TCP servers of our environment to port 20 and 21; then from a command line we added the same Web servers to port 80 and 443, by entering the following two commands:

```
ndcontrol server add 9.24.104.105:80:9.24.104.44+9.24.104.127+9.24.104.227
ndcontrol server add 9.24.104.105:443:9.24.104.44+9.24.104.127+9.24.104.227
```

Coming back to the GUI, click the **Refresh** button in the top menu bar and you will get the configuration updated, as shown in the following figure:

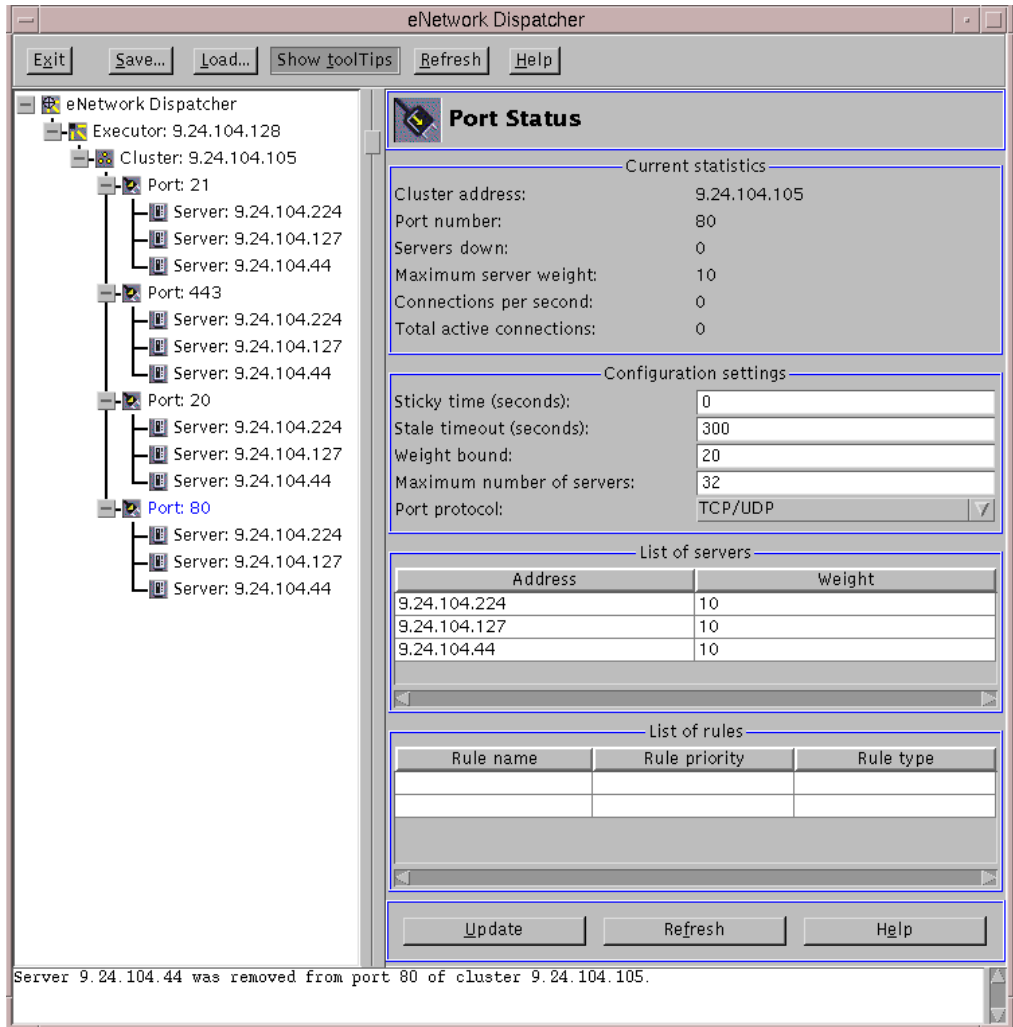


Figure 211. All Servers Added

At this point of the configuration, the Dispatcher machine is already capable of performing load balancing using *weighted round-robin scheduling*, which is based on the current servers' weights.

In general, weights are set by the Manager component of the eNetwork Dispatcher machine. The Manager component decides each single weight basing its decision on internal counters in the Executor, feedback from the advisors and feedback from a system-monitoring program, such as ISS. Notice that weights are applied to all servers on a given port.

In this particular configuration, we wanted to set weights manually, and so we did not run the Manager component. We accepted the default values for weights. As you can see in Figure 211 on page 266, each server has a weight of 10 on the HTTP port 80. In this case, since all the servers would be balanced with the same weight, the weights would not impact the standard round-robin mechanism, so we implemented a simple non-weighted round-robin.

It is important to mention that the field *Weight bound*, visible in Figure 211 on page 266, can be used to set how much difference there can be between the number of requests each server will get. We did not modify the default value, 20,

which means that one server could get twenty times as many requests as another.

4.6.4 TCP Servers Configuration

Before the Dispatcher starts to forward TCP/IP connection requests to the TCP servers, it is necessary to set up the TCP server machines. Precisely, on each server in the cluster, you must add an alias to the cluster address on the loopback interface, often called lo0. The loopback IP address is usually 127.0.0.1 and is never forwarded as a destination on the network media.

OS/2 and HP-UX Operating Systems

If you have a server running on an operating system that does not support aliases, such as HP-UX and OS/2, you must *set* the loopback device to the cluster address.

Other operating systems, such as Windows NT, AIX and Solaris support aliases, so you can follow the procedure we describe now.

The Dispatcher does not change the destination IP address in the TCP/IP packet before forwarding the packet to a TCP server machine. By setting or aliasing the loopback device to the cluster address, the TCP server machine will accept a packet that was addressed to the cluster address.

Before going on with the TCP server's configuration, you should understand what the flow of the incoming and outgoing IP packets is. The next section provides detailed information about that. However, if you just want to go on with the configuration steps, you can skip this section and go directly to 4.6.4.1, "Aliasing the Loopback Device on AIX" on page 267.

4.6.4.1 Aliasing the Loopback Device on AIX

To alias the loopback device on an AIX machine, use the following command syntax:

```
ifconfig lo0 alias cluster_address netmask netmask
```

For example, on our AIX Web server machine (see Table 4 on page 246), we entered the command:

```
ifconfig lo0 alias 9.24.104.105 netmask 255.255.255.0
```

You must alias the loopback device each time your AIX machine reboots, so it is recommended that you put the previous command in a script and instruct the machine to run it at reboot time. What we did in our case was to put the previous command into the file /etc/rc.tcpip. This file contains commands that start TCP/IP daemons (sendmail, inetd, etc.) on an AIX machine. It is launched at each reboot. We added at the end of that file the following two lines. (Note, however, that the first line only contains a comment.)

```
#Add alias on the loopback device: it is needed when using with eND  
ifconfig lo0 alias 9.24.104.105 netmask 255.255.255.0
```

4.6.4.2 Aliasing the Loopback Device on Windows NT

The procedure to add the alias on a Windows NT system is more complex, because you have to first add and then configure the loopback device, as reported in the following.

Click **Start** then **Settings**, and select **Control Panel**. Double-click on the **Network** icon, and in the Network window click on the **Adapters** tab. Then select the **Add...** button and from the list of adapters select **MS Loopback Adapter**, as shown in the following window:

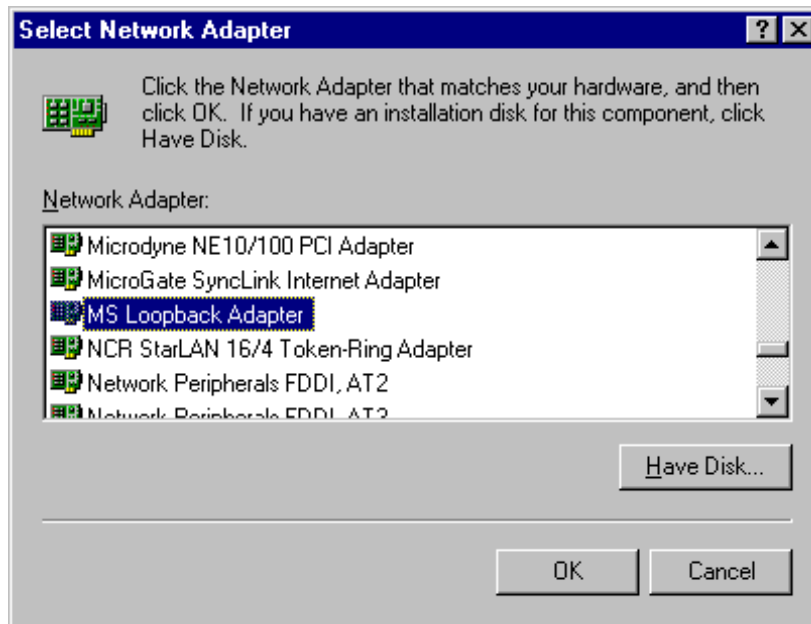


Figure 212. Select Network Adapter: MS Loopback Adapter

Click **OK** and you will get the following screen:

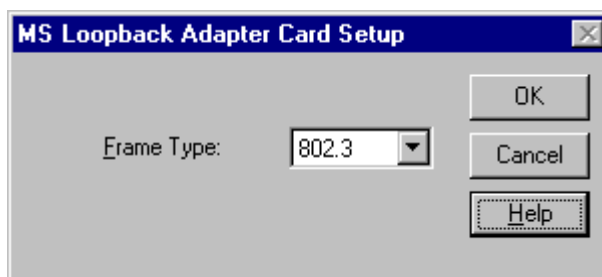


Figure 213. MS Loopback Adapter Card Setup

We accepted the default configuration, clicked **OK** and, when prompted, inserted the installation Windows NT CD-ROM.

After these simple steps, you will see another adapter added in your Windows NT machine adapter list:

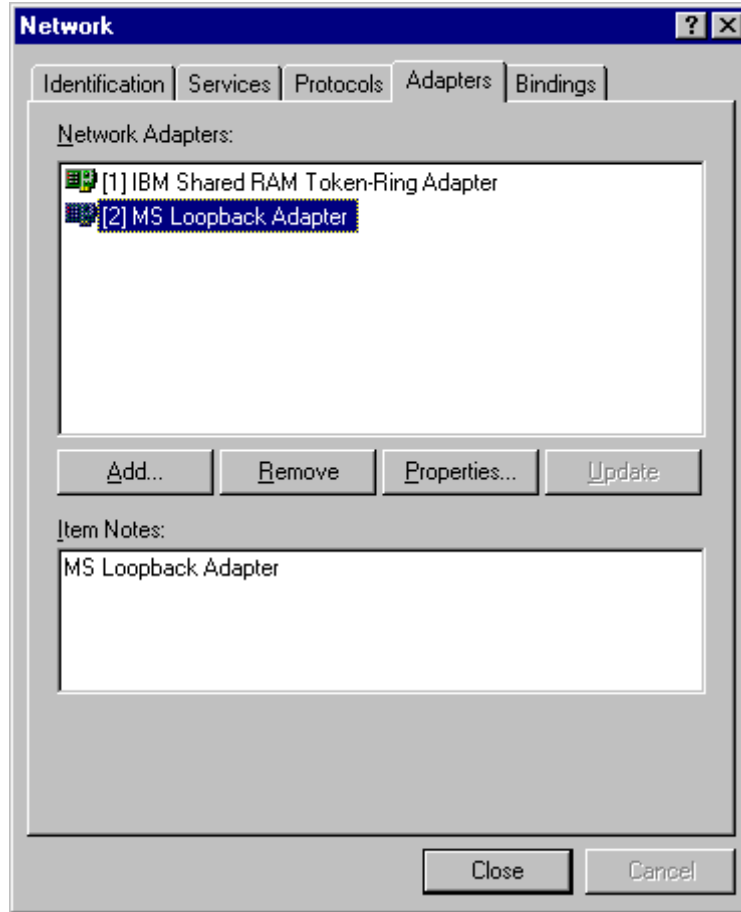


Figure 214. Adapters List Refreshed

Now select the **Protocols** tab and double-click on the **TCP/IP Protocol** item. If you open the Adapter list box, the MS Loopback Adapter should have been added to the list.

Note

If the MS Loopback Adapter does not appear in the Adapter list box under the TCP/IP protocol configuration window, this is because the Network panel needs to be refreshed. You need to close and re-open it to see it updated with the new adapter.

Select **MS Loopback Adapter** from the Adapter list box and set the IP Address to the cluster address. We accepted the default subnet mask 255.0.0.0 and did not enter any gateway address, as shown in the following figure:

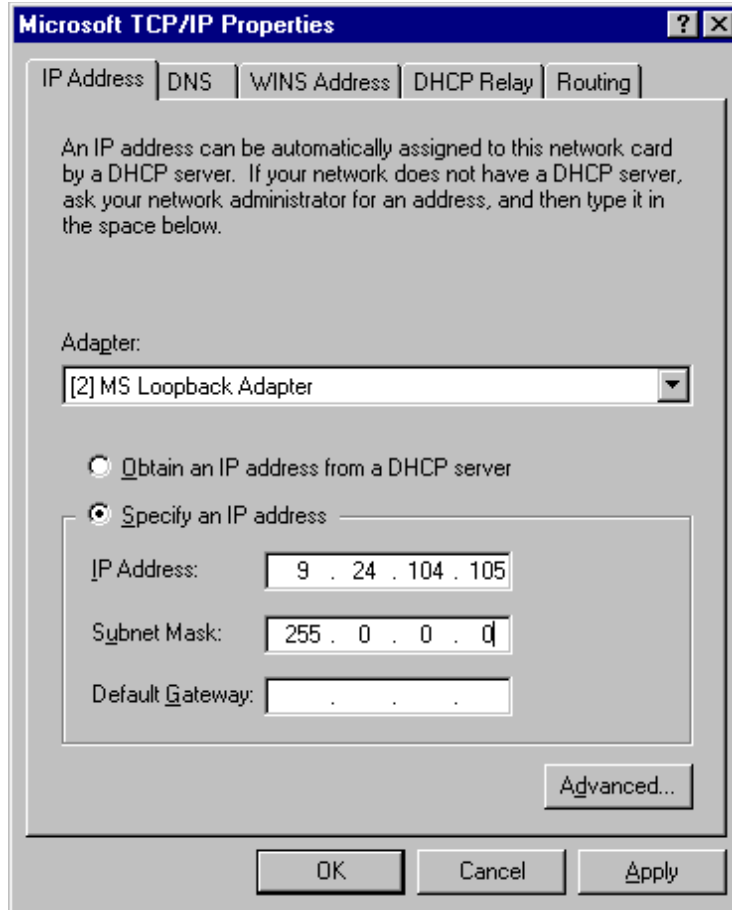


Figure 215. Configuring MS Loopback Adapter

Upon selecting the **OK** button, you will get the following window:

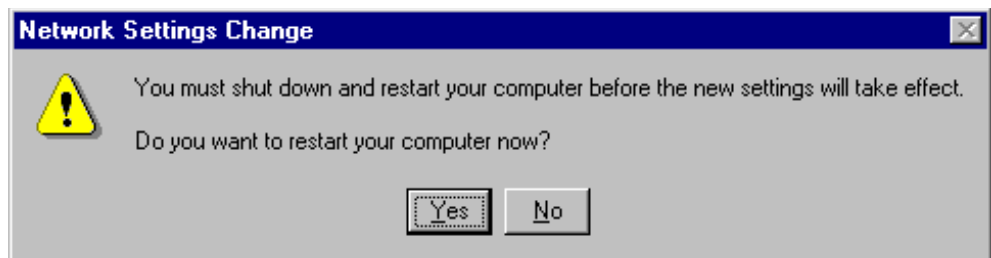


Figure 216. Network Setting Change - Reboot Your System

Choose **OK** to reboot your system.

On Windows NT, after you install the MS Loopback Adapter, each time you reboot the system you do not need to redefine the alias on your loopback device. Instead, you have to face the drawback that an extra route is created at each reboot, and you need to delete it.

To check for an extra route, from a command prompt enter the command:

```
route print
```

A table similar to the following will be displayed:

```

Command Prompt
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\>route print

Active Routes:

Network Address      Netmask    Gateway Address  Interface    Metric
0.0.0.0              0.0.0.0    9.24.104.1      9.24.104.44  1
9.0.0.0              255.0.0.0  9.24.104.105    9.24.104.105  1
9.24.104.0           255.255.255.0  9.24.104.44    9.24.104.44  1
9.24.104.1           255.255.255.255  9.24.104.44    9.24.104.44  1
9.24.104.10          255.255.255.255  9.24.104.44    9.24.104.44  1
9.24.104.44          255.255.255.255  127.0.0.1      127.0.0.1    1
9.24.104.105         255.255.255.255  127.0.0.1      127.0.0.1    1
9.24.104.239         255.255.255.255  9.24.104.44    9.24.104.44  1
9.255.255.255        255.255.255.255  9.24.104.44    9.24.104.44  1
127.0.0.0            255.0.0.0  127.0.0.1      127.0.0.1    1
224.0.0.0            224.0.0.0  9.24.104.105    9.24.104.105  1
224.0.0.0            224.0.0.0  9.24.104.44    9.24.104.44  1
255.255.255.255     255.255.255.255  9.24.104.44    9.24.104.44  1

C:\>

```

Figure 217. Extra Route

In order to find the extra route, you should look for your cluster address under the Gateway Address column. If the cluster address appears twice, it means you have an extra route. In our case, the cluster address 9.24.104.105 appeared in row 2 and row 11.

You need only one of those two entries; one of them is *extra*, and must be removed. To understand what the extra route is between the two that show up in the above table, follow this simple rule: the extra route is the one whose address, in the Network Address column, begins with the first number of the cluster address, followed by three zeros. In our case, the extra route is the one in row 2, since it has network address 9.0.0.0.

Table 6. Extra Route

9.0.0.0	255.0.0.0	9.24.104.105	9.24.104.105	1
---------	-----------	--------------	--------------	---

We entered the following command to delete the extra route:

```
route delete 9.0.0.0 9.24.104.105
```

Then we checked that the extra route had been deleted by entering the following command as shown in Figure 218 on page 272:

```
route print
```

```

C:\>route delete 9.0.0.0 9.24.104.105
C:\>route print
Active Routes:
    Network Address        Netmask    Gateway Address  Interface    Metric
    0.0.0.0                0.0.0.0    9.24.104.1      9.24.104.44  1
    9.24.104.1            255.255.255.0  9.24.104.44    9.24.104.44  1
    9.24.104.10          255.255.255.255  9.24.104.44    9.24.104.44  1
    9.24.104.44          255.255.255.255  127.0.0.1      127.0.0.1    1
    9.24.104.61          255.255.255.255  9.24.104.44    9.24.104.44  1
    9.24.104.69          255.255.255.255  9.24.104.44    9.24.104.44  1
    9.24.104.105        255.255.255.255  127.0.0.1      127.0.0.1    1
    9.24.104.108        255.255.255.255  9.24.104.44    9.24.104.44  1
    9.24.104.192        255.255.255.255  9.24.104.44    9.24.104.44  1
    9.24.104.239        255.255.255.255  9.24.104.44    9.24.104.44  1
    9.255.255.255        255.255.255.255  9.24.104.44    9.24.104.44  1
    127.0.0.0            255.0.0.0    127.0.0.1      127.0.0.1    1
    224.0.0.0            224.0.0.0    9.24.104.105   9.24.104.105  1
    224.0.0.0            224.0.0.0    9.24.104.44    9.24.104.44  1
    255.255.255.255     255.255.255.255  9.24.104.44    9.24.104.44  1

```

Figure 218. Extra Route Deleted

In order to automatically remove the extra route at each system reboot, you can copy the mentioned command in a batch text file, named for example Routedel.bat, and place it into your Windows NT Startup folder, as shown:

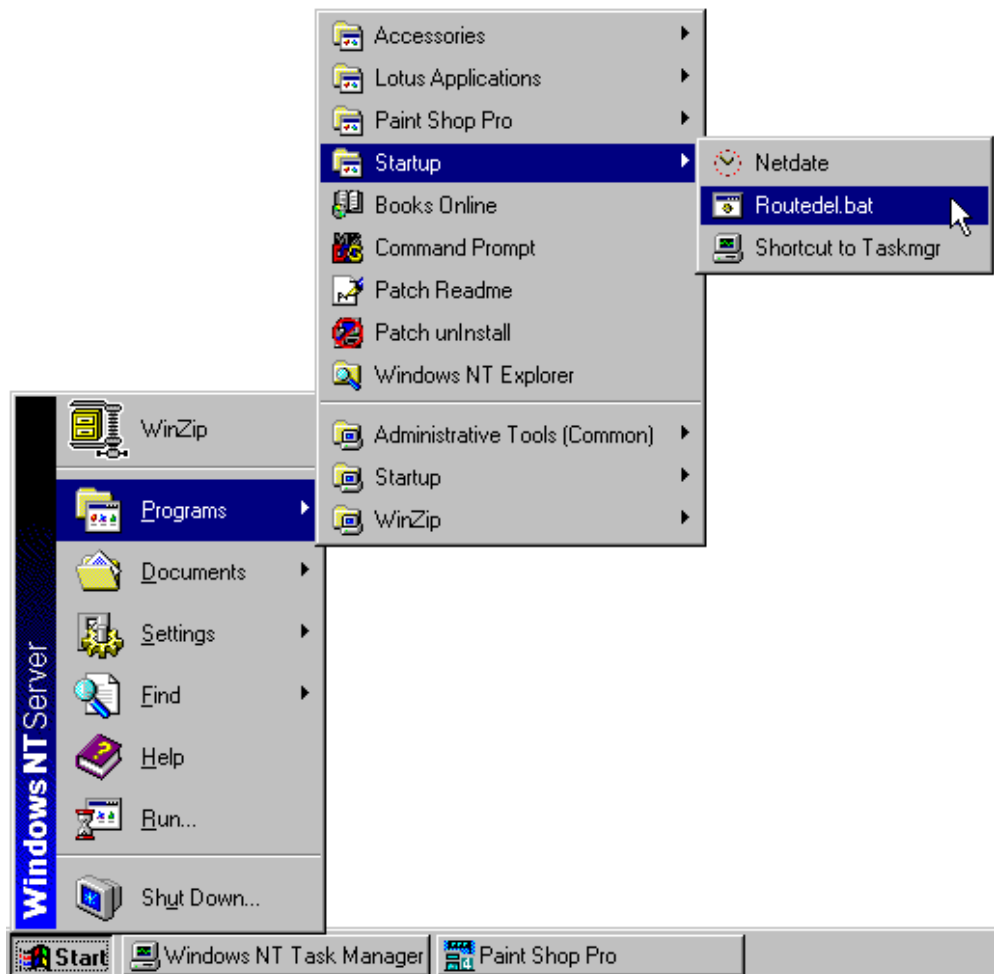


Figure 219. Startup Folder of Windows NT Server

4.6.5 How the Dispatcher Works: The Flow of the IP Packets

In 4.3, “How the Dispatcher Function Works” on page 213, we have explained how the IP packets start flowing from a TCP client to a TCP server passing through the Dispatcher machine, and then how the server’s response flows directly from the server to the client without any need to pass through the Dispatcher again. In this section, we want to explain more details on how this important mechanism works.

You need to issue the IP configuration on the Dispatcher machine and on the TCP servers. You also need to define aliases to the cluster address on the network interface on the Dispatcher machine and on the loopback device on all the cluster’s TCP servers. The reason for this is based on the way the Dispatcher works and treats incoming IP packets.

The Dispatcher is designed to make several TCP servers appear as one in the TCP/IP environment, typically for HTTP, FTP, and other protocols on the Internet.

Let’s assume that a request for the service managed by the cluster arrives. The incoming IP packets have, among other data: the source address, which identifies who has sent the packet, and the destination address, which is the IP address of the cluster. For example, in the environment we built, an incoming IP packet has 9.24.104.239 its source IP address, since this was the address of our Web client machine, and 9.24.104.105 as the destination IP address, since this was the cluster address (see Table 4 on page 246).

When the Dispatcher is installed, all the incoming IP packets sent by end users to the cluster address first arrive at the Dispatcher machine, not at one of the TCP servers. This is because the Dispatcher’s network interface, besides having its own unique IP address, has been given an alias to the cluster address. The TCP servers in the cluster also have an alias to the cluster address, but this is defined on the loopback interface. Other things to keep in mind is that the Dispatcher runs at a low level in the machine operating system so it can directly intercept all the IP packets.

Each time a new connection is initiated by a client, the Dispatcher selects which TCP server in the cluster should receive the connection packet. Now the Dispatcher should be able to send that packet to the selected TCP server.

How can the Dispatcher do that? The answer is that the Dispatcher routes the packet based on the MAC address of the network adapter on the chosen TCP server.

Media Access Control (MAC) Address

Ethernet and token-ring devices require an adapter to physically attach to the LAN. This adapter must provide both physical and logical capabilities for the device. The adapter contains a unique 48-bit address, assigned to it during the manufacturer process, called Media Access Control (MAC). All the MAC addresses are assigned by the IEEE 802 committee. The IEEE provides the vendor building adapters with a range of MAC addresses to use for assigning adapters their unique 48-bit address so that no two adapters should ever have a duplicate address.

Ethernet and token-ring require the MAC address for both the origin and the destination adapters when communicating over a LAN. Besides the IP address, the MAC address also must be known when sending data to a LAN-attached device.

So, let's assume that the Dispatcher has decided to send the connection IP packet to the TCP server that has the following (see Table 4 on page 246):

- Unique IP address as 9.24.104.224 and host name computer1
- The cluster address 9.24.104.105 as the alias on the loopback interface
- Adapter MAC address equal to 0004AC9693EE.

The Dispatcher must now transmit the IP packet over the token-ring network (but Ethernet would be similar), so it must know the MAC address of the selected TCP server.

Address Protocol Request (ARP)

If the TCP server's MAC address is unknown to the Dispatcher, the same Dispatcher is able to discover it by issuing an Address Protocol Request (ARP) containing a broadcast LAN MAC address that will be read by all the network interfaces attached to the same LAN. This request also specifies the IP address 9.24.104.244 of the TCP server. Each of the network adapters attached to the LAN will determine if its relative interface has been configured with the IP address 9.24.104.224. Only the network interface adapter of computer1 will respond with its own MAC address, 0004AC9693EE.

Once the Dispatcher knows the wanted TCP server's MAC address, it will be stored into the ARP cache to skip successive requests.

The original destination MAC address on the IP packet was the one of the network interface on the Dispatcher machine itself. When the Dispatcher knows the MAC address of the selected TCP server computer1, it changes the original destination MAC address on the packet to the MAC address 0004AC9693EE. This packet will be now encapsulated in the token-ring frame and transmitted to the chosen TCP server computer1.

The Dispatcher then sets up a connection table entry to make sure that subsequent incoming IP packets for this client continue being forwarded to the same server.

When the TCP server computer1 receives the packet, the information related to the MAC addresses is eliminated and the source and destination IP addresses are extracted.

The destination IP address is still the cluster address 9.24.104.105, but the TCP server computer1 has its own IP address 9.24.104.224. However, computer1 can anyway accept that packet, since the cluster address is configured as an alias on the computer1's loopback interface.

At this point, computer1 sends a response to the Web client that originated the request. Now, let's see what happens to the outgoing packets.

Once the TCP server computer1 receives all the IP packets of the originating client's request, it performs the standard TCP processing that is commonly performed by all TCP servers while responding to a client. It switches the IP source and destination addresses for the outgoing packets that form the response to the client. The source address becomes in this case the cluster address 9.24.104.105, while the destination address is now the Web client IP address 9.24.104.224. This operation has an important consequence; the balancing function is transparent both to the client and the clustered servers.

1. The destination IP address is the client's IP address, not the Dispatcher's. For this reason, the TCP server computer1 can route the IP packets through its default router directly to the client, and all the outgoing packets do not pass back through the Dispatcher. There is no need to even return using the original physical path and a separate high-bandwidth connection can be used.

This is very important, since in many cases, the volume of the outbound server-to-client traffic is substantially greater than the inbound traffic. For example, HTML pages and imbedded images are at least 10 times the size of the client URLs that requested them.

2. The source IP address in the outgoing packet sent by the TCP server shows as the cluster address, and not as the TCP server's IP address. The cluster address was also the destination IP address in the IP packets sent by the client in its request, so the client is not able to understand the TCP architecture of the target server. This security feature offered by the Load Balancing component ensures privacy for a site that is composed by multiple TCP servers load-balanced by a Dispatcher machine.

Because the Dispatcher does not participate in bidirectional communications with the client but simply forwards the incoming packets unchanged, its presence is transparent to both client and server. The real TCP/IP connection is between the client and the clustered server, and the Dispatcher soon disappears from the scene after forwarding the incoming packets.

The only requirement for the TCP server is that its loopback device be set or aliased to the cluster address. In this way, the server is capable of responding to a request that was addressed to the cluster address.

4.6.6 Round-Robin Load Balancing Scenario

Now that the TCP servers are configured also, we want to show you how the Dispatcher is able to make standard round-robin load balancing. Before going in-depth, it is worth to give an explanation about the weights of the servers connecting to a port in the cluster. As we also mentioned in 4.6.3, "Dispatcher

Configuration” on page 248, weights are applied to all servers on a given port, and for any particular port, the requests will be distributed between servers based on their weights relative to each other. For example, if the weight on one server is set to 10 and on the other server to 5, the first server gets twice as many requests as the second server.

In our configuration, since the weight for each server on port 80 had been set to 10 (see Figure 211 on page 266), we expected the Dispatcher to perform the standard (not weighted) round-robin load balancing.

To do that, we performed the following actions. From a command line on the Dispatcher machine, we entered the following command:

```
ndcontrol server report 9.24.104.105:80:9.24.104.127+9.24.104.44+9.24.104.224
```

We got the total number of TCP connections on port 80 to the servers:

```

dtterm
Window Edit Options Help
# ndcontrol server report 9.24.104.105:80:9.24.104.127+9.24.104.44+9.24.104.224
-----
Cluster: 9.24.104.105 Port: 80
Address      | Total | TCP | UDP | Active | FINned | Complete | SaWt
-----|-----|-----|-----|-----|-----|-----|-----
9.24.104.127|    0 |  0 |  0 |    0 |    0 |    0 |   -1
-----|-----|-----|-----|-----|-----|-----|-----
9.24.104.44 |    0 |  0 |  0 |    0 |    0 |    0 |   -1
-----|-----|-----|-----|-----|-----|-----|-----
9.24.104.224|    0 |  0 |  0 |    0 |    0 |    0 |   -1
-----|-----|-----|-----|-----|-----|-----|-----
#

```

Figure 220. TCP Connections on Port 80

We could verify, in this way, that no connection had yet been activated.

Then we wrote three HTML files, which were very simple and very similar to another. The three of them were named enzo.html, and we distributed them all into the home directories of the three clustered Web servers of our platform:

```

<HTML>
<!-- enzo.html on host 9.24.104.44 (Windows NT Server 4.0) -->
<!-- copied into D:\WWW\HTML -->

<TITLE>WTR05219</TITLE>
<BODY>
This page comes from wtr05219.itso.ral.ibm.com, having IP adress
9.24.104.44
</BODY>
</HTML>

```

Figure 221. D:\WWW\HTML\enzo.html on Host 9.24.104.44

```

<HTML>
<!-- enzo.html on host 9.24.104.224 (Windows NT Server 4.0) -->
<!-- copied into D:\WWW\HTML -->

<TITLE>COMPUTER1</TITLE>
<BODY>
This page comes from computer1.itso.ral.ibm.com, having IP address
9.24.104.224
</BODY>
</HTML>

```

Figure 222. D:\WWW\HTML\enzo.html on Host 9.24.104.224

```

<HTML>
<!-- enzo.html on host 9.24.104.127 (AIX 4.3.1) -->
<!-- copied into /usr/lpp/internet/server_root/pub-->

<TITLE>COMPUTER1</TITLE>
<BODY>
This page comes from rs600022.itso.ral.ibm.com, having IP adress
9.24.104.127
</BODY>
</HTML>

```

Figure 223. /usr/lpp/internet/server_root/pub/enzo.html on Host 9.24.104.127

Notice that the three files have the same structure. In each file it is specified what server in the cluster is actually the owner of the page. This will be displayed not only in the body of the Web page, but also in the title bar of the browser window, since we have inserted the host name of the TCP server between the HTML tags `<TITLE>` and `</TITLE>`.

We started our Web browser and de-activated its memory and disk caches; then we requested the URL `http://9.24.104.105/enzo.html` and got the following screen:

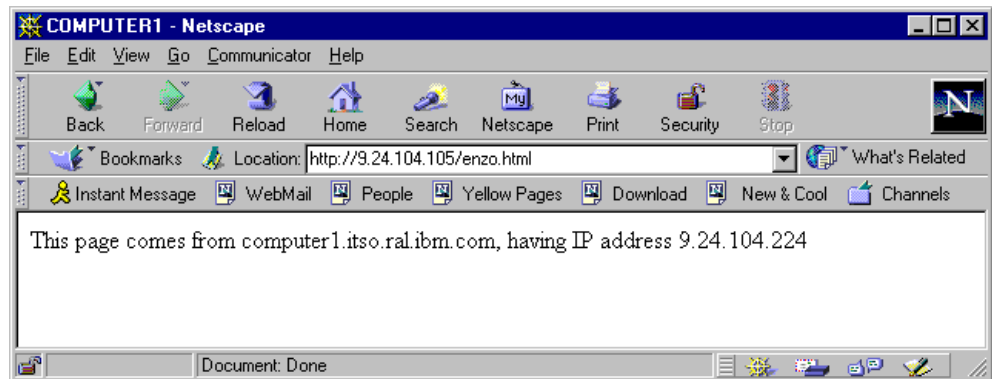
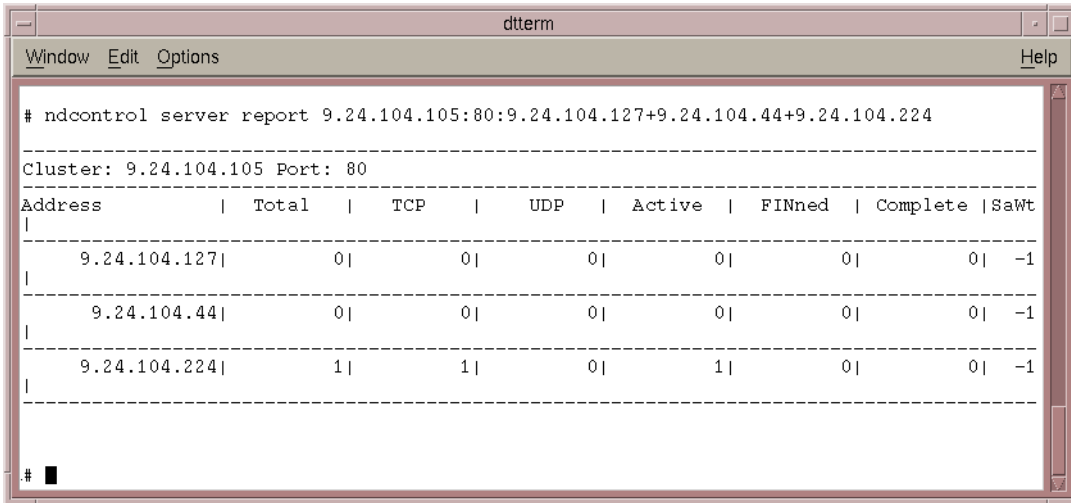


Figure 224. HTML Page Served by the First Web Server

Note that the requested page came from the Web server with IP address 9.24.104.224. By entering the following command again we saw the Dispatcher report:

```
ndcontrol server report 9.24.104.105:80:9.24.104.127+9.24.104.44+9.24.104.224
```

It confirmed that the server with IP address 9.24.104.224 had effectively served the request, as shown in the following figure:



```
# ndcontrol server report 9.24.104.105:80:9.24.104.127+9.24.104.44+9.24.104.224
```

Cluster: 9.24.104.105 Port: 80

Address	Total	TCP	UDP	Active	FINned	Complete	SaWt
9.24.104.127	0	0	0	0	0	0	-1
9.24.104.44	0	0	0	0	0	0	-1
9.24.104.224	1	1	0	1	0	0	-1

```
.#
```

Figure 225. The First Web Server Has Served One Page

Then we re-loaded the page from the browser by clicking the **Reload** button. This time the page came from another server, the one with IP address 9.24.104.127, as shown in the following figure:

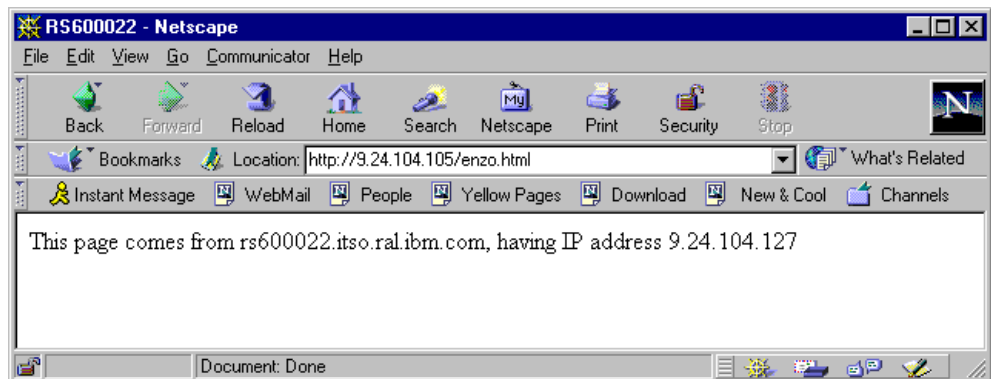


Figure 226. HTML Page Server by the Second Web Server

Upon re-entering the following command we had another confirmation that the server with IP address 9.24.104.127 had effectively served the request:

```
ndcontrol server report 9.24.104.105:80:9.24.104.127+9.24.104.44+9.24.104.224
```

This confirmation is shown in the following figure:

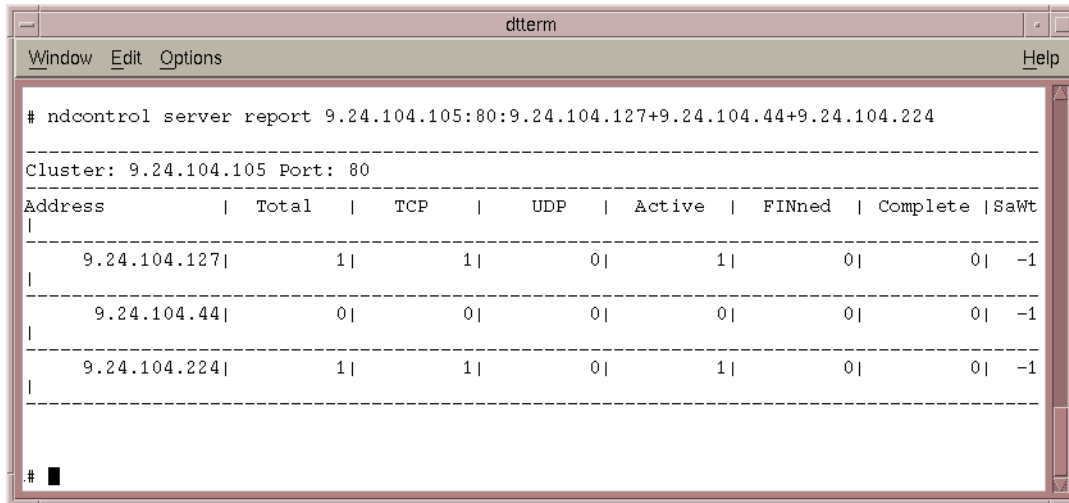


Figure 227. The Second Web Server Has Served One Page

At the end, we re-loaded the page on the browser for the third time and saw that the file enzo.html was served by the server with IP address 9.24.104.44, as shown in the following figure:

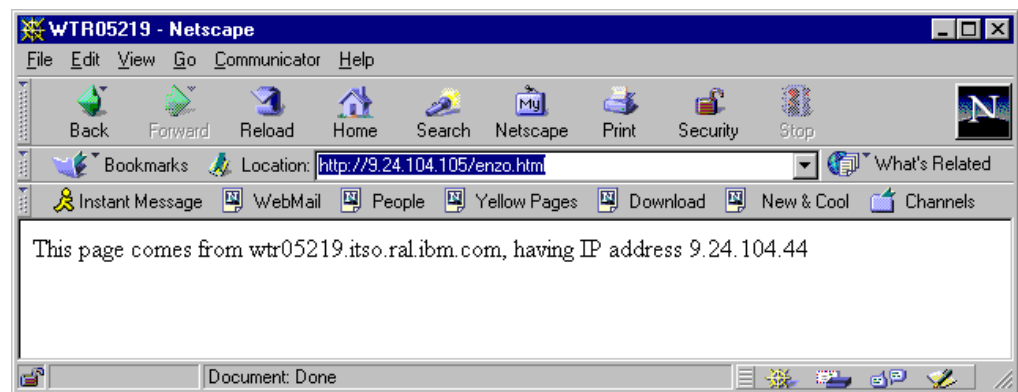


Figure 228. HTML Page Served by the Third Server

We entered again the command:

```
ndcontrol server report 9.24.104.105:80:9.24.104.127+9.24.104.44+9.24.104.224
```

This is what we saw:

```

dtterm
Window Edit Options Help
# ndcontrol server report 9.24.104.105:80:9.24.104.127+9.24.104.44+9.24.104.224
-----
Cluster: 9.24.104.105 Port: 80
-----
Address      | Total | TCP | UDP | Active | FINned | Complete | SaWt
-----
 9.24.104.127|    1 | 1 | 0 | 0 | 1 | 0 | -1
-----
 9.24.104.44 |    1 | 1 | 0 | 1 | 0 | 0 | -1
-----
 9.24.104.224|    1 | 1 | 0 | 1 | 0 | 0 | -1
-----
# █

```

Figure 229. The Third Web Server Has Served One Page

So each time you reload the same Web page from the browser, the servers honoring the response alternate in a non-weighted round-robin way.

It is worth emphasizing that we used a different test page on each of the three servers. Each Web page was named enzo.html, but the contents were slightly different. We did this just to show the round-robin balancing of the Dispatcher. In a real situation, the pages would be identical, and this could be obtained by either duplicating the same Web page in all the Web servers or using the File Sharing component of IBM WebSphere Performance Pack to dynamically distribute the same file in multiple locations.

When using the Load Balancing component of IBM WebSphere Performance Pack, if the page is really the same, the user cannot see any differences, since even the URL in the Location field of the browser and the information displayed on the Page Info window of the browser do not show that the page is served by different Web servers. The cluster address of your site is the only IP address that is shown. In this sense, the load balancing operation is transparent to the end user and even to the clustered Web server, since it can respond directly to the client, as explained in 4.6.5, “How the Dispatcher Works: The Flow of the IP Packets” on page 273.

4.6.7 Analyzing the Flow with a Network Monitoring Tool

In this section we describe an experience that lets us understand better how the Dispatcher function really works. In this experience, we repeated the scenario described in 4.6.6, “Round-Robin Load Balancing Scenario” on page 275, but on the client machine we had Microsoft Network Monitor Version 4.00 for Windows NT Server running. This package can capture and display incoming packets being transmitted from other machines within the same LAN segment where the server running Network Monitor is attached.

The host name of the client machine was wtr05193, as specified in Table 4 on page 246. From the Web browser installed on this machine, we sent some requests to the clustered Web site we implemented, having cluster address

9.24.104.105, and we monitored the incoming network packets. We got the following table from the Network Monitor:

Frame	Time	Src MAC Addr	Dst MAC Addr	Protocol	Description	Src Other Addr	Dst Other Addr	Type Other Addr
21	11.175	00062968264E	WTR05193	TCP	.A.P...., len: 220, seq: 11522	9.24.104.105	WTR05193	IP
22	11.197	0004AC9693EE	WTR05193	TCP	.A..S., len: 4, seq: 670170	9.24.104.105	WTR05193	IP
23	11.284	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670170	9.24.104.105	WTR05193	IP
24	11.284	0004AC9693EE	WTR05193	TCP	.A.P...., len: 8, seq: 670172	9.24.104.105	WTR05193	IP
25	11.286	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670172	9.24.104.105	WTR05193	IP
26	11.290	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670173	9.24.104.105	WTR05193	IP
27	11.291	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670175	9.24.104.105	WTR05193	IP
28	11.292	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670176	9.24.104.105	WTR05193	IP
29	11.296	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670178	9.24.104.105	WTR05193	IP
30	11.298	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670179	9.24.104.105	WTR05193	IP
31	11.299	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670180	9.24.104.105	WTR05193	IP
32	11.300	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670182	9.24.104.105	WTR05193	IP
33	11.301	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670183	9.24.104.105	WTR05193	IP
34	11.303	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670185	9.24.104.105	WTR05193	IP
35	11.303	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670186	9.24.104.105	WTR05193	IP
36	11.372	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670188	9.24.104.105	WTR05193	IP
37	11.373	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670189	9.24.104.105	WTR05193	IP
38	11.374	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670191	9.24.104.105	WTR05193	IP
39	11.434	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670192	9.24.104.105	WTR05193	IP
40	11.435	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670193	9.24.104.105	WTR05193	IP
41	11.436	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670195	9.24.104.105	WTR05193	IP
42	11.543	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670196	9.24.104.105	WTR05193	IP
43	11.544	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670198	9.24.104.105	WTR05193	IP
44	11.545	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670199	9.24.104.105	WTR05193	IP
45	11.573	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670201	9.24.104.105	WTR05193	IP
46	11.573	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670202	9.24.104.105	WTR05193	IP
47	11.574	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670204	9.24.104.105	WTR05193	IP
48	11.594	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670205	9.24.104.105	WTR05193	IP
49	11.595	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670207	9.24.104.105	WTR05193	IP
50	11.596	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670208	9.24.104.105	WTR05193	IP
51	11.617	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670209	9.24.104.105	WTR05193	IP
52	11.618	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670211	9.24.104.105	WTR05193	IP
53	11.619	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670212	9.24.104.105	WTR05193	IP
54	11.640	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670214	9.24.104.105	WTR05193	IP
55	11.640	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670215	9.24.104.105	WTR05193	IP
56	11.642	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670217	9.24.104.105	WTR05193	IP
57	11.661	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670218	9.24.104.105	WTR05193	IP
58	11.662	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670220	9.24.104.105	WTR05193	IP
59	11.663	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670221	9.24.104.105	WTR05193	IP
60	11.693	0004AC9693EE	WTR05193	TCP	.A...., len: 1452, seq: 670223	9.24.104.105	WTR05193	IP

Figure 230. Using the Network Monitor

In this table, take a look at the columns Src Other Addr and Dst Other Addr. You can see that the client machine is not able to realize which server has honored the HTTP requests, since the source address from which the response arrived is always the cluster address (see 4.6.5, “How the Dispatcher Works: The Flow of the IP Packets” on page 273).

However, if you look closely at the table displayed by the Network Monitor, you can see that Frame 21 (and all the other frames before that) was served by the clustered Web server whose network interface had MAC address 00062968264E, while starting from Frame 22 the Web server having MAC address 0004AC9693EE started serving. This confirmed the analysis shown in 4.6.5, “How the Dispatcher Works: The Flow of the IP Packets” on page 273.

Notice that the experience we have just described would have not worked if the TCP servers had been located on a different LAN. In other words, we would not have been able to see the MAC addresses of the various TCP servers if such Web servers had been located on a different LAN. For this reason, in a real-life situation, the TCP architecture of a clustered Web site remains hidden to remote clients, which results in a further security feature offered by the Load Balancing component of IBM WebSphere Performance Pack.

4.6.8 Activating the Manager and the Advisors

Now let's go back to the GUI on the eNetwork Dispatcher machine and activate the load balancing components responsible of the server performances: the Manager and the Advisors.

4.6.8.1 Activating the Manager

The main load balancing component is the Manager. The Manager is the component that collects the information from the Advisors about the servers' condition. Based on that information, it then adjusts the weights of the single server to reconfigure load distribution.

If you want to start the Manager component through the GUI, right-click with your mouse on the **eNetwork Dispatcher** item (see Figure 211 on page 266) and from the pop-up menu select **Start Manager...** You are prompted to optionally type a log file name, in which the manager will log all its activities, and a metric port used by the ISS function, if running, to report system loads. We set Log File name to `eNDmanager1` and we left blank the Metric port field, as shown in the following figure:

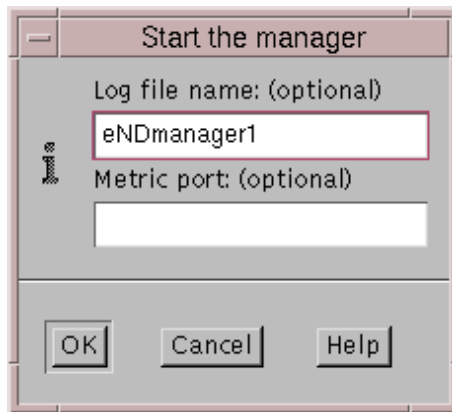


Figure 231. Starting the Manager

In 4.3.4, "TCP Ports Used by the Dispatcher" on page 217, we have already explained how to start the Manager function through the command line. Leaving the Metric port field blank forces the system to use the default value 10004.

4.6.8.2 Activating the Advisors

In order to feed the manager with more information about the ability of the TCP servers to respond to requests, you need to start the Advisors. As we explained in 4.3.1, "Dispatcher Components" on page 214, the Advisors monitor each server defined on the assigned port, and forward the information about the server's response time and availability to the Manager.

In the following table you find the list of the available Advisors along with their respective protocols and ports:

Table 7. Advisors and Related Protocol and Ports

Advisor Name	Protocol	Port
ftp	FTP	21
telnet	Telnet	23

Advisor Name	Protocol	Port
smtp	SMTP	25
http	HTTP	80
pop3	POP3	110
nnntp	NNTP	119
ssl	SSL	443

Note that if you use the ftp advisor, it should be started only on the FTP control port 21, and not on the FTP data port 20 (see also Table 5 on page 261).

To start the advisor from the GUI, right-click with your mouse on **Manager** and select **Start Advisor...** from the pop-up menu that is automatically brought up, as shown in the following figure:

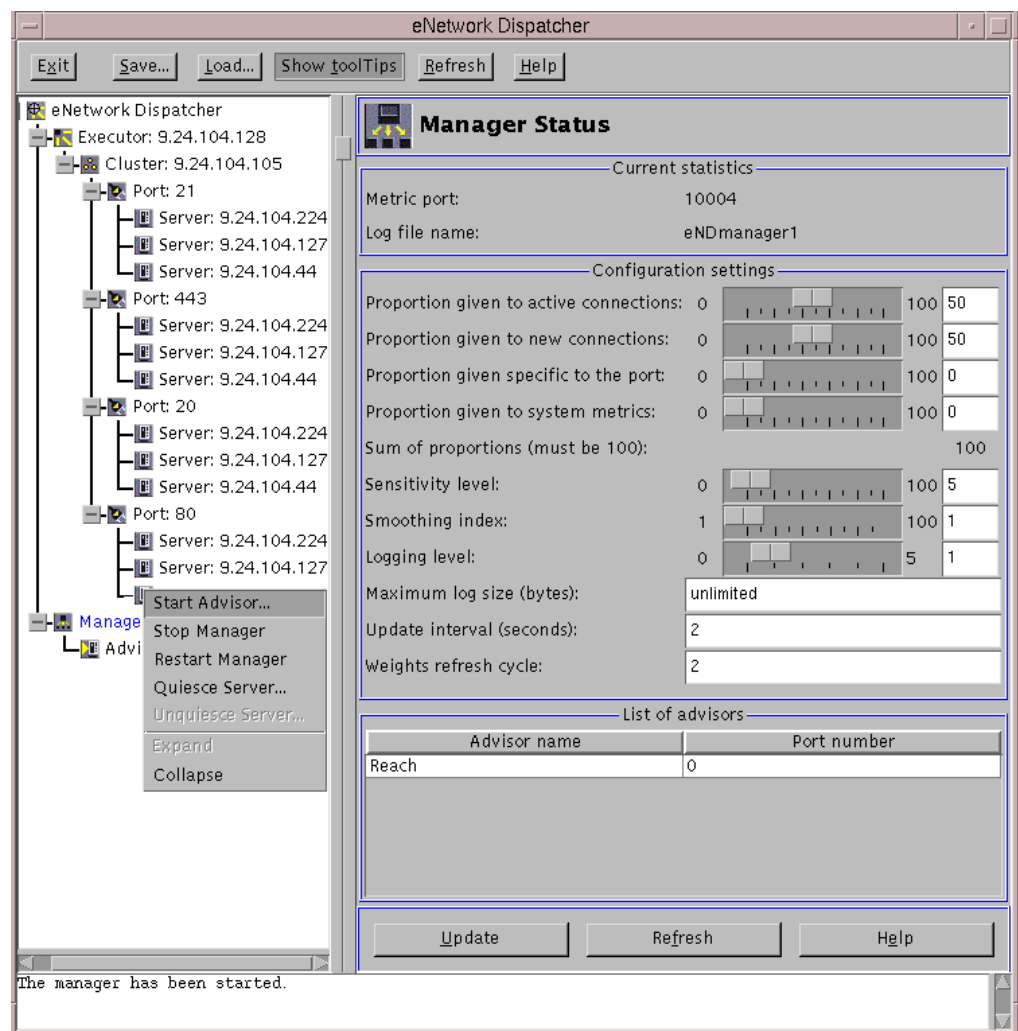


Figure 232. How to Start the Advisor

Select the advisor you want to add from the list box in the following panel. We wanted to select the ssl advisor, so we chose **Ssl** and the port was automatically set to 443, as shown in the following figure:



Figure 233. Select Advisor

If you click **OK**, you will see that the Advisor has been added to the configuration, as shown in the following panel:

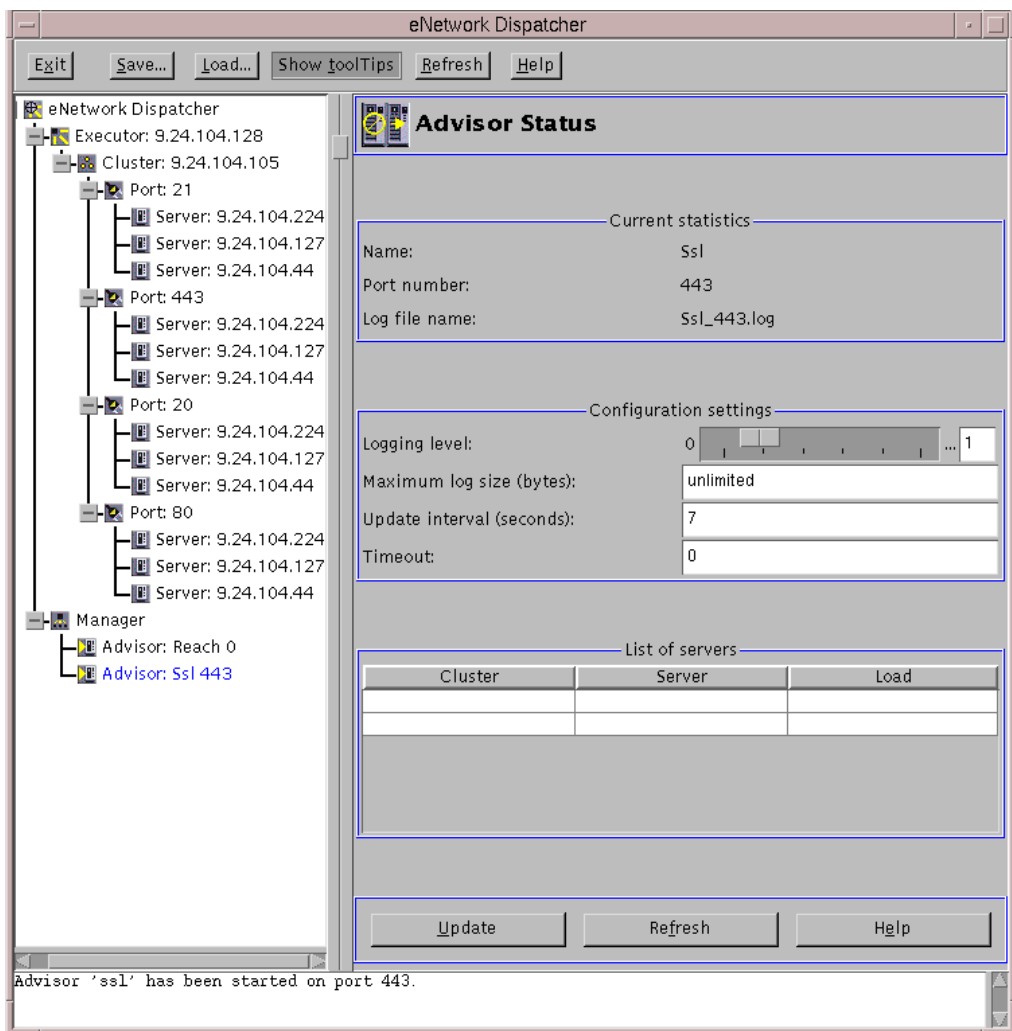


Figure 234. Advisor Added

We also added the http and ftp Advisors, but for these we wanted to experiment with the command line, so we entered the following two commands:

```
ndcontrol advisor start http 80
ndcontrol advisor start ftp 21
```

Then we went back to the GUI, selected the **Refresh** button on the top menu bar and we had the configuration updated, as shown in the following figure:

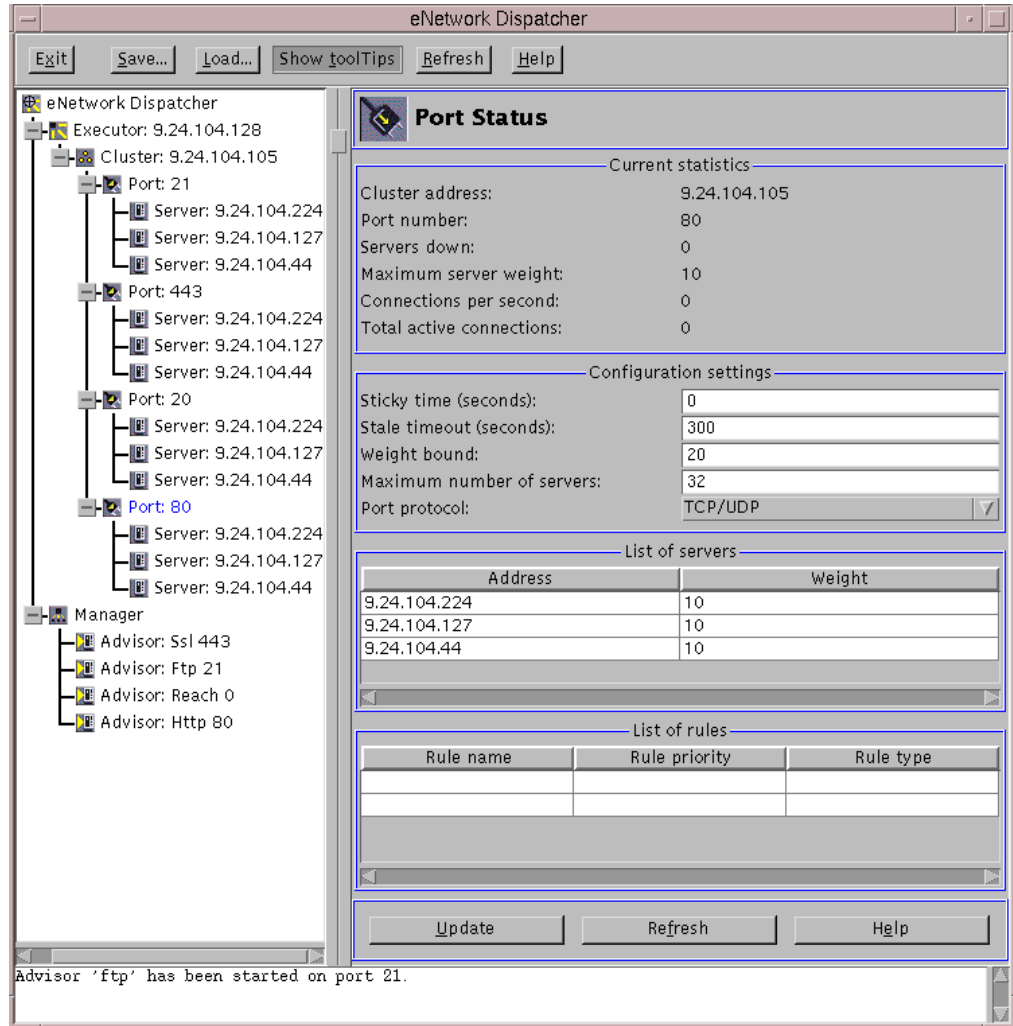


Figure 235. Other Advisors Added

4.6.9 Customization of the Manager and the Advisors

The load balancing action performed by the Dispatcher depends on several settings, which we are going to describe in the following sections.

4.6.9.1 Proportions of Importance Settings

Now we show you how to set the proportions of importance for each of the policies (see 4.3.2.1, “Guidelines on Proportions of Importance Settings” on page 215). In the configuration window shown in Figure 235 on page 285, click on the **Manager** item in the tree panel and on the right window the Manager Status window will appear. As we said in 4.3.2, “Proportions of Importance” on page

215, the default values are 50 50 0 0: we tried the mix 46 46 8 0, as shown in Figure 236 on page 287, so that:

- Both the number of active connections and new connections will contribute 46% in the weighting process.
- The advisors will contribute 8%.
- Input from system monitoring tools, such as ISS, will not be considered in the weight decision. This is reasonable, because in this test we had not yet activated any monitoring tool.

4.6.9.2 Smoothing Index

Another important parameter to be considered when using the Advisors (and/or system monitoring tools such as ISS) is the *smoothing index*, which defines how smooth the change of the servers' weight will be. The Manager calculates and then updates the servers' weights dynamically. The weight values could change very rapidly, and in some circumstances this might result in an *oscillant* effect in the way the requests are load balanced. The distribution of the requests could be made in a non-optimized way, and needs to be smoothed. Based on the smoothing index value, the Manager will change the servers' weights more or less quickly. The default value for smoothing index is 1.5, which could cause the Manager to change the weights rather dynamically. Values around 5 are preferred for having the Manager modify the weights slowly.

We set the Smoothing index value to 6. After you modify all the values you want in the Manager Status window, click **Update** to have the configuration updated, as shown in the following window:

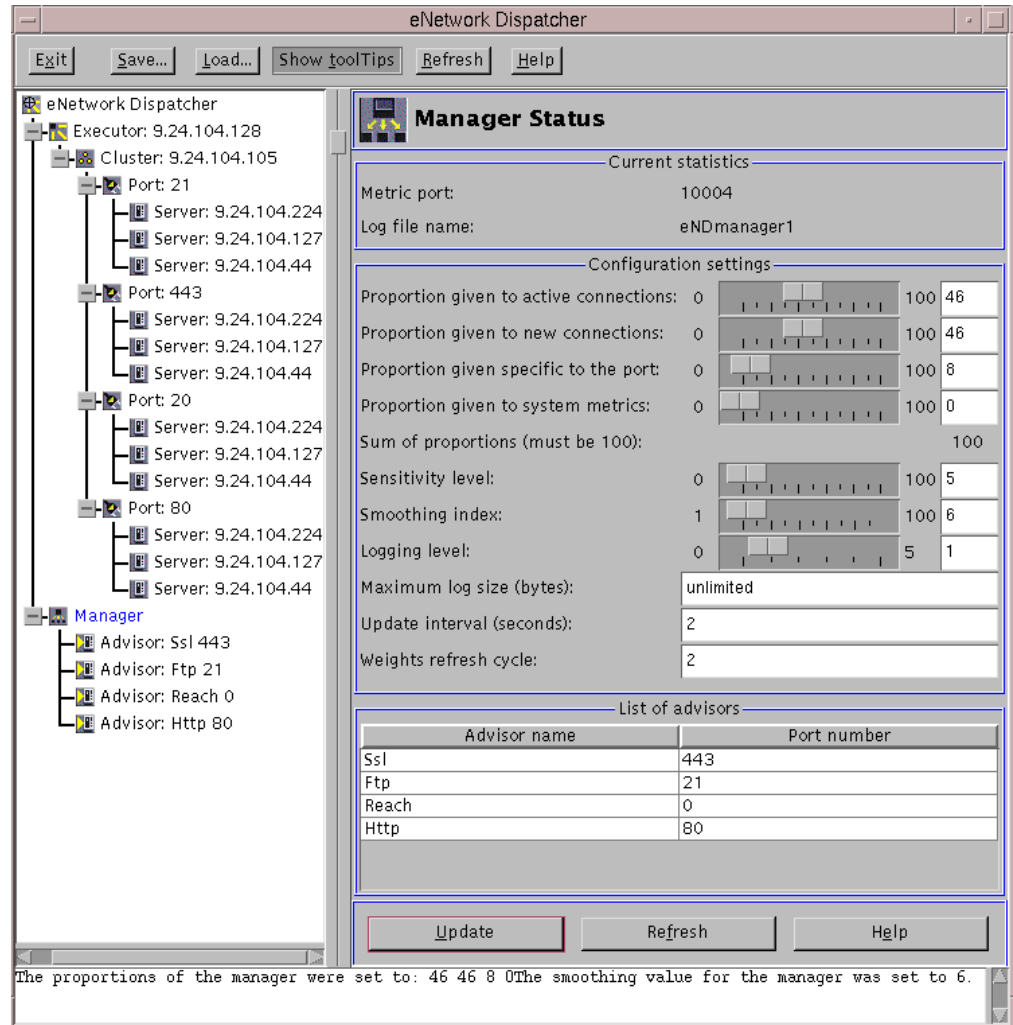


Figure 236. Setting the Manager Proportions and Smoothing Level

4.6.10 Saving the Configuration

Once you finish the configuration steps we have described, it is recommended you save the configuration of the Load Balancing machine. To do so, click the **Save...** button in the top menu bar of the configuration window (see Figure 236 on page 287) and in the panel that appears, enter the configuration file name, as shown next:

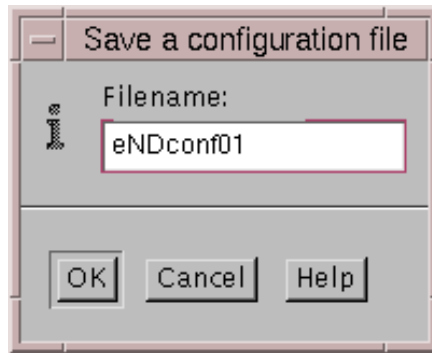


Figure 237. Enter Configuration File Name

We entered `eNDconf01`. When you click **OK**, if the platform for your eNetwork Dispatcher machine is AIX, the configuration file will be saved in the location `/usr/lpp/eND/dispatcher/configurations`. On our Windows NT platform, the directory where the configuration files got saved was `D:\Program Files\eND\dispatcher\Configurations`.

The configuration file `eNDconf01` that we saved is an ASCII file. We show it in the following figure:

```

ndcontrol server add 9.24.104.105:21:9.24.104.224
ndcontrol server set 9.24.104.105:21:9.24.104.224 weight 0

ndcontrol port add 9.24.104.105:20

ndcontrol server add 9.24.104.105:20:9.24.104.44
ndcontrol server set 9.24.104.105:20:9.24.104.44 weight 9

ndcontrol server add 9.24.104.105:20:9.24.104.127
ndcontrol server set 9.24.104.105:20:9.24.104.127 weight 9

ndcontrol server add 9.24.104.105:20:9.24.104.224
ndcontrol server set 9.24.104.105:20:9.24.104.224 weight 9

ndcontrol port add 9.24.104.105:443

ndcontrol server add 9.24.104.105:443:9.24.104.44

ndcontrol server add 9.24.104.105:443:9.24.104.127

ndcontrol server add 9.24.104.105:443:9.24.104.224

ndcontrol manager start eNDmanager1
ndcontrol manager proportions 46 46 8 0
ndcontrol manager smoothing 6.0

ndcontrol advisor start Http 80 Http_80.log

ndcontrol advisor start Ssl 443 Ssl_443.log

ndcontrol advisor start Ftp 21 Ftp_21.log

```

Figure 238. Configuration File eNDconf01

To stop the Executor, open the eNetwork Dispatcher GUI and right-click with your mouse on the **Executor** item in the left area. Then select **Stop Executor** from the pop-up menu that is brought up (see Figure 203 on page 259). Otherwise, if you chose to use the command line, enter the following:

```
ndcontrol executor stop
```

To exit the GUI, click the **Exit** button.

On Closing the GUI

Remember that if you only close the GUI without stopping the Executor, you have the Executor already running. In fact, if you re-start the GUI with the command `ndadmin`, you will still be shown the current running configuration.

Once you stop the Executor, if you start the GUI again, you can choose whether to load a saved configuration, if any, or create a new one. To load a configuration file, click the **Load...** button in the top menu bar and select a file name from the list box, as shown in the following panel:

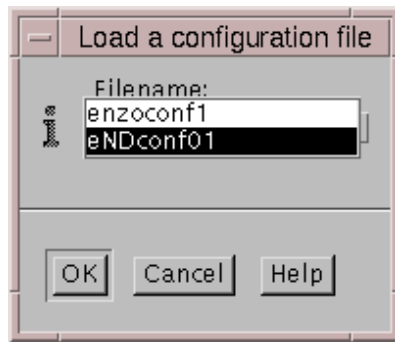


Figure 239. Loading a Configuration File

At this point, the Dispatcher starts performing load balancing without using input from any system monitoring tools. By using the monitoring capabilities provided by ISS on the TCP server machines, you can provide the Dispatcher with load server information. In this case the ISS co-operates with the Dispatcher, but ISS does not make any load balancing decision. The ISS monitor collects specific server information such as CPU usage, memory usage and disk activity from the ISS agents running on the individual servers, and forwards it to the Dispatcher. The Dispatcher uses this load information, along with other sources of information, to determine which is the least loaded server of the cluster and then performs load balancing.

4.7 Installation of the Interactive Session Support Function

As we have already mentioned, the Load Balancing component of IBM WebSphere Performance Pack has in turn two subcomponents: the Dispatcher and the Interactive Session Support (ISS).

The ISS function can be installed separately from the other component, the Dispatcher. We have explained, in fact, that ISS has the special function of system monitoring. If you want ISS to be part of your environment, you should install and configure a machine to run as an ISS monitor, while on the TCP servers in the cluster an ISS agent process should run.

On our environment shown in Figure 195 on page 247 we selected to have the Dispatcher machine run as the ISS monitor too. On that machine, ISS had already been installed as part of the Load Balancing component of IBM WebSphere Performance Pack, so we did not need to perform any further installation steps on it.

Then we had to install and configure ISS on the three TCP servers. On those machines, the ISS agent process would run, gathering information about the status of the systems and feeding that information back to the ISS monitor.

In this section we illustrate how you can install the ISS function on the AIX and Windows NT platforms. The installation on Sun Solaris is not very different from the installation on AIX. We recommend that for further details you refer to *eNetwork Dispatcher for Solaris, Windows NT and AIX User's Guide*, GC31-8496.

12. When the command completes, you can exit by pressing the F10 functional key.

To make sure that the product has been installed correctly, from a command line enter the command:

```
lsipp -h | grep intnd
```

If you have successfully installed the product (U.S. English version) the following lines will be returned:

```
intnd.iss.license  
intnd.iss.rte  
intnd.msg.en_US.iss
```

4.7.2 Installation on Windows NT

The software and hardware platform we used for this installation is described in 4.5.2, "Installation on Windows NT" on page 235. We performed the installation of ISS also on other machines with 64MB of RAM.

If you see again 4.5.2, "Installation on Windows NT" on page 235, you will find how the installation program for the Load Balancing component of IBM WebSphere Performance Pack can be performed via the Java InstallShield's setup class. We followed those directions until you were requested to choose the component to install; since we wanted to install only ISS, we selected only the two items **Session Support Program Files** and **Session Support Program Files**, as shown in the next window:

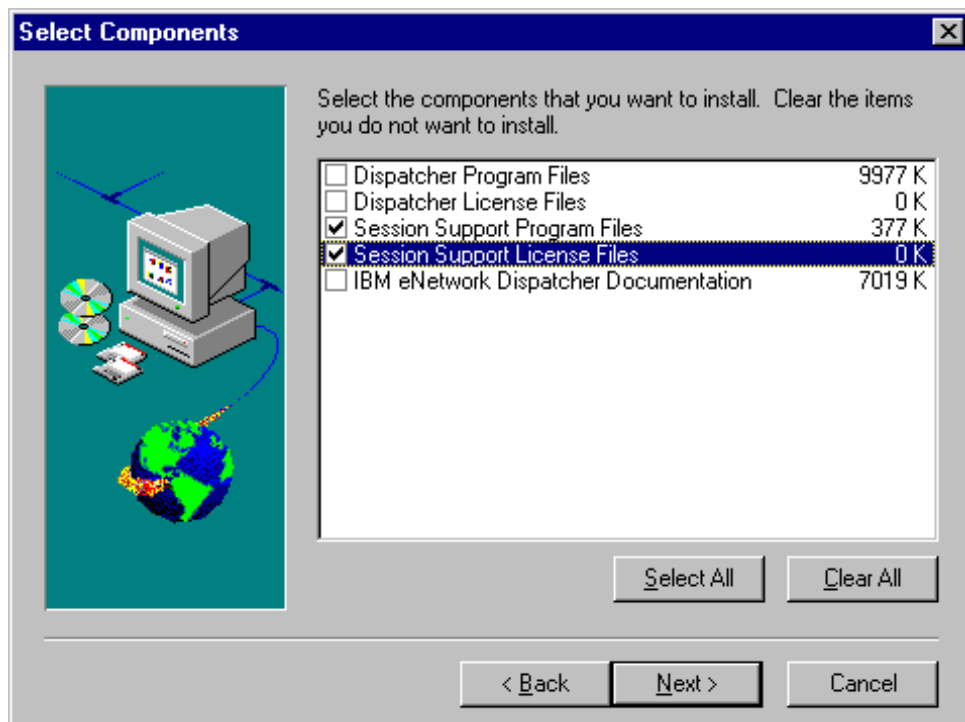


Figure 241. Selecting ISS Components

To continue, choose **Next**. The dialog for the selection of the language files is shown:

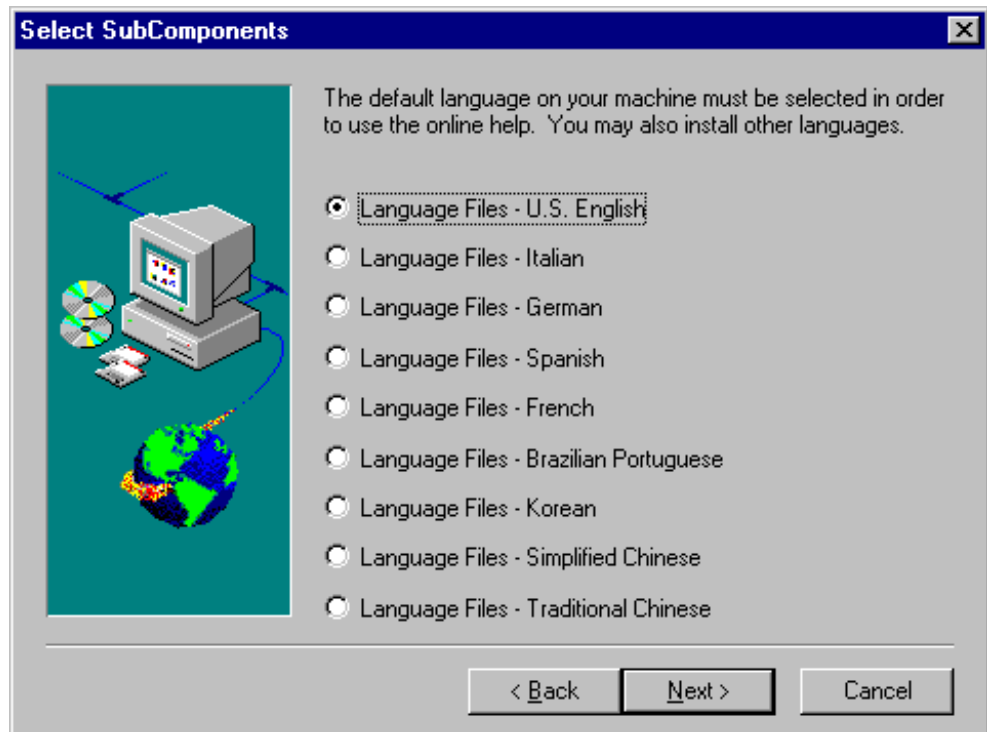


Figure 242. Select Language Files Window

We selected **Language Files - U.S. English**. As soon as you click on **Next**, such a window disappears and the installation procedure goes on and completes, but without giving back any other signal. We did not receive any notification that the installation was completed and no panel was brought up to suggest we reboot our system, as usually happens after a complex installation.

You can realize that the installation is finished if you check the services available on your computer in the Services dialog box of the Control Panel folder: you will see that a new service named `IBM_ISS_Load_Balancing` is in the list of all the available services of Windows NT. Notice that after the installation this service has not yet started, since its Startup mode is set to Manual rather than to Automatic, as you can see in the following screen:

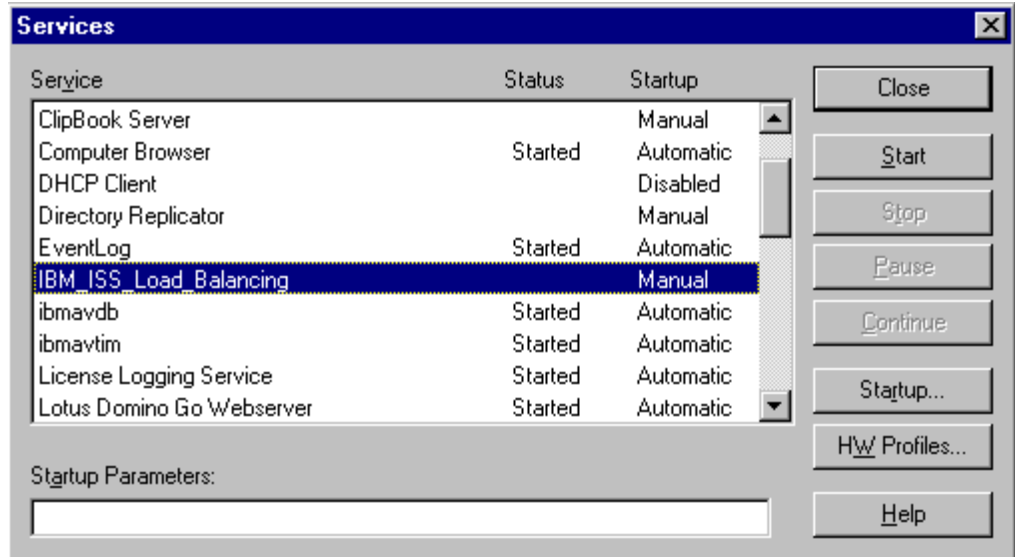


Figure 243. IBM_ISS_Load_Balancing Service Available on Windows NT

Even if the installation process did not require it explicitly, we decided to restart the machine after this installation.

4.8 Load Balancing Scenario Using the Dispatcher and ISS Functions

In this section we show you how to create a scenario where both the Dispatcher and ISS functions of the Load Balancing component are used. ISS is used here to gather server load information from the TCP servers, where the ISS agent process would run. This information is then passed back to the ISS monitor process, running on the same Dispatcher machine. The monitor interacts with the Dispatcher and provides it with information about the loads on the servers.

We used the same environment shown in Table 4 on page 246. However, we enhanced that scenario by adding the ISS function. The following figure offers a graphical representation of the environment we used:

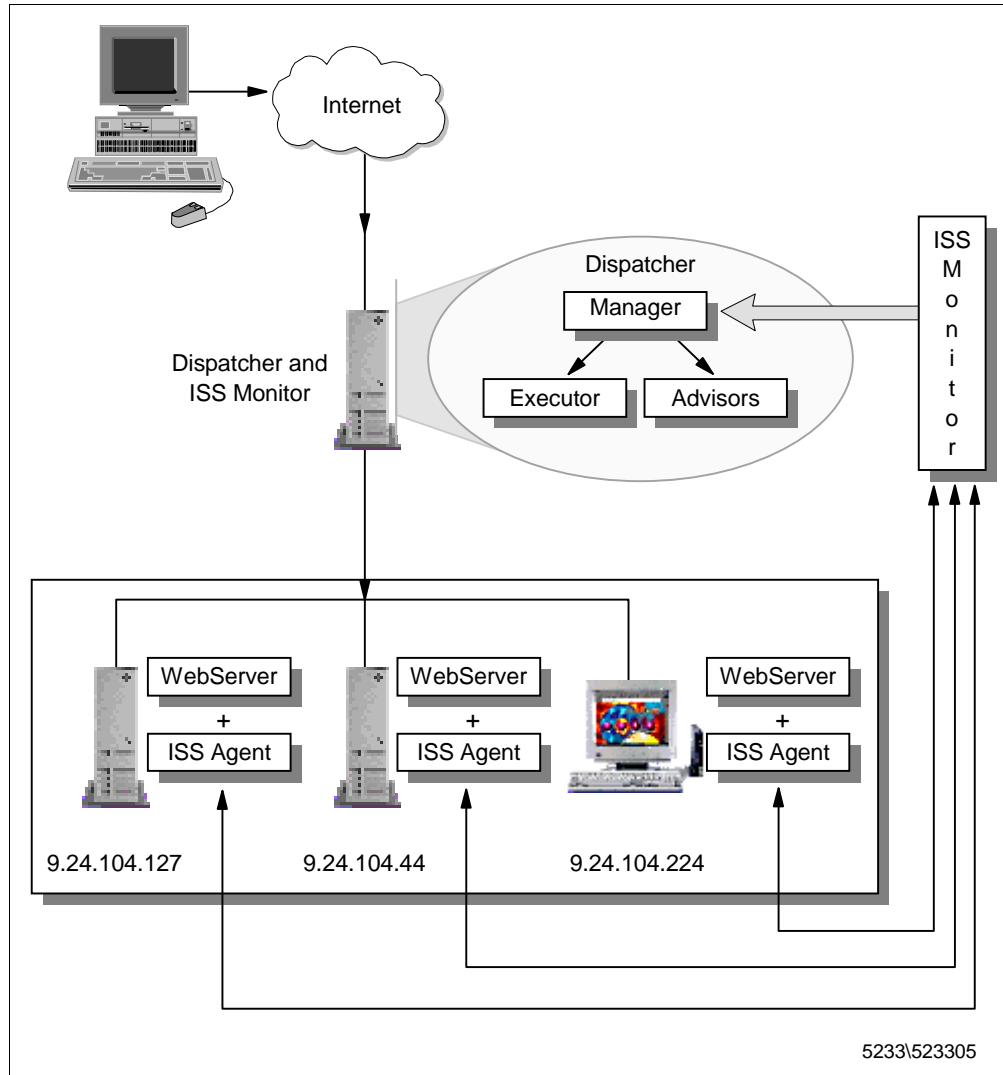


Figure 244. Graphical Representation of the Scenario Environment

Notice that, as soon as you make ISS part of the load balancing process as we are showing now, the Manager configuration must reflect the presence of ISS. In fact, it makes sense now that the proportions of importance (see 4.3.2, “Proportions of Importance” on page 215) take in account the input coming from ISS. In this case, we set the proportions of importance as 48 48 2 2 (see also 4.3.2.1, “Guidelines on Proportions of Importance Settings” on page 215).

The example we are going to show you is also very interesting because it demonstrates how easy it is to implement ISS high availability.

4.8.1 ISS Configuration File

In 4.6, “Load Balancing Basic Scenario Scenario Using the Dispatcher” on page 246, we described the network environment we had set up using the Dispatcher subcomponent, but without using ISS. Then we wanted to experiment with ISS, and we added the ISS subcomponent to the already existing environment. In this section we show and comment the ISS configuration file related to the sample network environment we set up. To build this file, we made some changes to the

sample configuration file `Iss_config_1` shipped with the product and located in the directory `D:\Program Files\eND\iss\bin\samples\en_US` on Windows NT, and `/usr/lpp/eND/iss/samples/en_US` on AIX.

The original `Iss_config_1` configuration file, as it is shipped with the product, is shown in Figure 170 on page 221 and Figure 171 on page 222 and we have already commented on it in 4.4.2, "ISS Configuration Files" on page 219.

The following figure shows the file we used on our platform, after modifying `Iss_config_1`. Notice that this file must be installed on all the machines that implement the ISS subcomponent, either monitors or agents. We got this file by modifying the sample configuration file `Iss_config_1`, and we renamed it as `Issbase.conf` on AIX and `Issbase.cfg` on Windows NT. The fact that the ISS configuration files carry such extensions is not mandatory. We just followed the convention that configuration files on AIX usually carry a `.conf` extension, while configuration files on Windows NT usually carry a `.cfg` extension. In the following, we refer to this file as `Issbase`, without any extension. What is important is that when you configure the ISS machines in your cell, all the ISS configuration files have the same contents.

```

# -----
# Sample ISS configuration file 1
# -----
# Configuration of a local cell
# This is a simple configuration file,
# with only one (local) cell, and one service running.
# Parameters for the whole cell
Cell      Hursley          local
AuthKey   10043572 ADE4F354 7298FAE3 1928DF54 12345678
LogLevel                      Error
HeartbeatInterval              5
HeartbeatsPerUpdate            3
PortNumber                      7139

# Individual node data
# Node numbers do not have to be sequential
# rs600023 is the monitor node
# rs600022 is the backup monitor node
# wtr05219 and computer1 are prevented from taking over the role of monitor.

Node      rs600023          001
Node      rs600022          002
Node      wtr05219          098
NotMonitor
Node      computer1         099
NotMonitor

# The service is only configured to depend on one resource -- CPU availability.
# Load balancing is therefore performed based only on CPU utilisation.
# However, ISS will not schedule work for nodes that are unreachable on the network.
# The specified MetricLimits indicate that a node will not be used
# if its CPU usage goes over 95% and will not be put back in the list
# until CPU usage goes back down to 80%.

ResourceType          CPU
Metric Internal       CPULoad
MetricNormalization  0      100
MetricLimits          80      95
Policy                Min

# The one configured service WWWService is an identifying string for the service,
# and is used when balancing between cells.
# The service DNS name is clusterend.itso.ral.ibm.com.
# followed by the cluster address and the port number.

Service      WWWService      clusterend.itso.ral.ibm.com 9.24.104.105  80
NodeList     rs600022 wtr05219 computer1
ResourceList CPU
SelectionMethod Best

# The machine rs600023.itso.ral.ibm.com has a
# Network Dispatcher configured on port 10004
Dispatcher   rs600023.itso.ral.ibm.com 10004
ServiceList  WWWService

```

Figure 245. ISS Configuration File *Issbase.conf*

If you compare the configuration file `Issbase` with `Iss_config_1`, shown in Figure 170 on page 221 and Figure 171 on page 222, you see some differences. We had made some modifications to the original file, as shown next:

- We set `LogLevel` to `Error` rather than to `Info`. This way, we instructed ISS to provide debugging data on errors occurring.
- We configured the Dispatcher machine `rs600023` to be also the primary ISS monitor, and the three Web servers `rs600022`, `computer 1` and `wtr05219` to be ISS agents. While `wtr05219` and `computer1` are explicitly set as non-monitors by using the `NotMonitor` keyword, `rs600022` was also backup to the monitor node. Notice in fact that the `NotMonitor` keyword is not specified for `rs600022` and that the priority numbers are set as `001` for `rs600023` and `002` for `rs600022`.

These simple steps are enough to implement ISS high availability.

- We modified the other fields in order to take in account the machines of our platform and the fact that the machine `rs600023` would act as a Dispatcher observer.

4.8.2 Managing ISS

This section explains how to start, control and stop the ISS function.

4.8.2.1 The `issd` Command

ISS makes use of only one daemon, named `issd`, which should be launched in each of the nodes belonging to a cell. The `issd` daemon is instructed on how to operate by the directions contained in the configuration file. For example, based on the parameter in the configuration file, the `issd` will operate as ISS monitor or ISS agent.

The `issd` daemon is launched through the `issd` command and can be used with different flags and parameters. The most important flags and parameters are `-c config_file` and `-l log_file`.

Use `-c config_file` to specify the configuration file. If not specified, `issd` should expect to find the file in a pre-defined default location, that is:

- `/etc/iss.cfg` on AIX
- `\Program Files\eND\iss\iss.cfg` on Windows NT

Notice that each node should have the same configuration file.

When you use `-l log_file`, the log information is directed, by default, to a predefined file:

- `/etc/iss.log` on AIX
- `\Program Files\eND\iss\iss.log` on Windows NT

Use this flag and parameter to use the specified `log_file` instead of the default.

You can also specify `-d` to debug the daemon startup process, `-h` to display a help message, `-i` to run the `issd` daemon interactively, and `-p port_number` to specify the number of port to use for control traffic.

4.8.2.2 Starting ISS on AIX

You can run the `issd` daemon in the foreground or background. To start the `issd` daemon, log in as root, enter `issd` and use any of the parameters as indicated in 4.8.2.1, “The `issd` Command” on page 298. In our case, we put our configuration file into location `/usr/lpp/eND/iss` and named it `Issbase.conf`. We wanted to also have the log file, named `Issbase.conf`, in the same directory. So the full command we entered was:

```
issd -c /usr/lpp/eND/iss/Issbase.conf -l /usr/lpp/eND/iss/Issbase.log
```

4.8.2.3 Starting ISS on Windows NT

Under Windows NT, the `issd` daemon runs as a Windows NT service, whose name is `IBM_ISS_Load_Balancing`. To start this service, log on as Administrator or as a user with administrative privileges, click **Start**, select **Settings**, then open the Control Panel window and double-click on **Services**. From the Services window select **IBM_ISS_Load_Balancing**, as shown in the following figure:

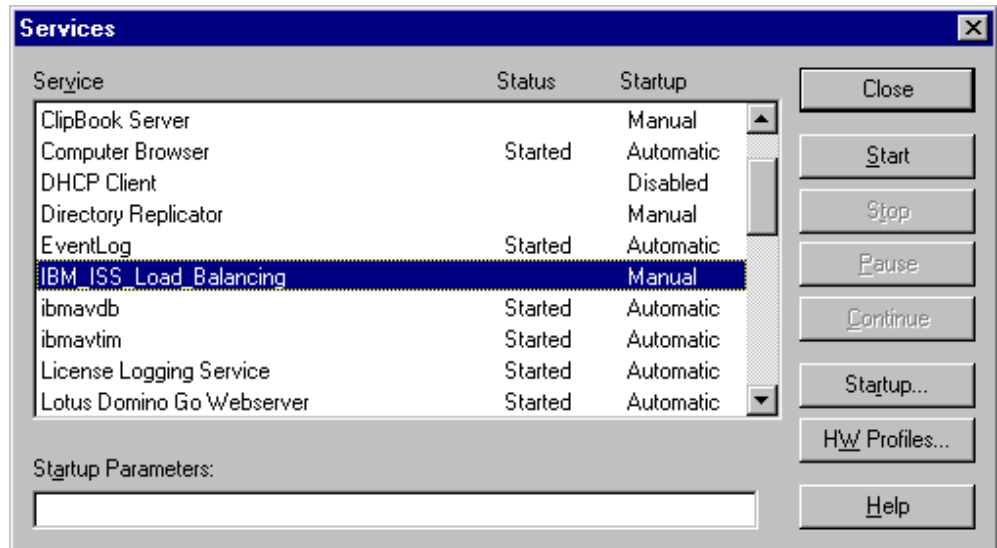


Figure 246. IBM_ISS_Load_Balancing Service on Windows NT

In the Startup Parameters text field you can specify the arguments for the `issd` command, as indicated in 4.8.2.1, “The `issd` Command” on page 298. After specifying the flags and parameters you need, click on the **Start** button.

If you want to code a backslash `\`, you must specify a double backslash `\\`. For example, in one of our tests the full name of the configuration file was `D:\issconf\issenzo1.cfg`, and the log file we chose to use was `D:\issconf\issenzo1.log`, so we typed:

```
-c D:\\issconf\\issenzo1.cfg -l D:\\issconf\\issenzo1.log
```

Notice that each time you stop and restart the service, you have to re-enter the Startup Parameters. Then, if you choose to automatically start the service at the reboot by setting its Startup type to **Automatic**, you can not decide the name and the path of your configuration and log files, but you must use the default, as indicated in 4.8.2.1, “The `issd` Command” on page 298.

4.8.2.4 Controlling ISS

You can control and re-configure ISS while it is running through the `isscontrol` command. Below we show an example of how you can use this command. All the details about the syntax of this command and the parameters it accepts are found in *eNetwork Dispatcher for Solaris, Windows NT and AIX User's Guide*, GC31-8496.

For example, while ISS was running in our cell, we wanted to add the node `rs600030`, specifying that it would be the third in line as monitor, and that it would belong to the service `WWWService`. From the command line, we entered the following two commands:

```
isscontrol add node rs600030 3
isscontrol add node rs600030 service WWWService
```

Then we needed to start routing requests to this new node. To do so, we entered:

```
isscontrol start node rs600030
```

As soon as the `isscontrol` command is issued on one of the machines running the `issd` daemon, it propagates to the other machines running the `issd` daemon, so that it should not be necessary to issue it on all the machines. However, we were informed that in Version 2.0 of the Load Balancing component there could be a problem with authorization that may prevent the above command from propagating properly, so you would have to run it from the other stations as well.

4.8.2.5 Stopping ISS

Here we explain how to stop ISS:

- On Windows NT, as administrator, from the Services window (see Figure 246 on page 299), select **IBM_ISS_Load_Balancing**, then click **Stop**. Such a button is enabled only if the service has already started.
- On AIX, log in as root and enter the command:

```
isscontrol shutdown node_name
```

where `node_name` is the name of the node on which you want to stop the `issd` daemon. For example, when we wanted to stop the `issd` daemon on the node `rs600022` of our cell, we entered:

```
isscontrol shutdown rs600022
```

4.9 Rule-Based Load Balancing

You can use rules-based load balancing to fine tune when and why packets are sent to which servers. The Dispatcher reviews any rules you add from first priority to last priority, stopping on the first rule that it finds to be true, and then balances the load between any servers associated with the rule. The Dispatcher already has the capability of balancing the load based on destination and port. Using rules expands your ability to distribute connections and offers more possibility to manage the load distribution in a cluster.

Rules are particularly useful when you want to distribute the load on a subset of your servers for any reason. You can use the following types of rules described in the following list:

- Client IP address

You might want to use rules based on the client IP address if you want to screen the customers and allocate resources based on where they are coming from.

For example, you have noticed that your network is getting a lot of unpaid and therefore unwanted traffic from clients coming from a specific set of IP addresses. You can add a rule that instructs the Dispatcher not to serve those requests, or to distribute them among a limited subset of your servers.

- Time of day

You may want to use rules based on the time of day for capacity planning reasons. For example, if your Web site gets hit most during the same group of hours every day, you might want to dedicate five servers to HTTP during full-time, then add another five during the peak time period.

Another reason you might use a rule based on the time of day could be, for example, if you want to bring some of the servers down for maintenance every night at midnight. In this case, you would set up a rule that excludes those servers during the necessary maintenance period.

- Connections per second on a port

You might want to use rules based on connections per second on a port if you need to share some of your servers with other applications.

For example, you set two rules:

1. If the number of connections per second on port 80 is greater than 100, then distribute the load on two specific Web servers.
2. If the number of connections per second on port 80 is greater than 2000 then distribute the load on 10 specific Web servers.

Note that the Manager must be running for the above to work.

- Active connections total for a port

You might want to use rules based on the total number of active connections on a specific port. This is very useful for when your servers get overloaded and start throwing packets away.

In fact certain Web servers will continue accepting connections even though they do not have enough threads to respond to the requests. As a result, the client requests time out and the customer coming to your Web site is not served. You can set rules based on the total number of active connections to balance capacity within a pool of servers.

For example, you know from your experience that your servers will stop serving requests after they have accepted 250 connections. You would create a rule that instructs the Dispatcher to use your current servers, but some additional servers should be added when the total number of active connections becomes greater than 250. Those additional servers will otherwise be used for other processing.

Note that the Manager must be running for the above to work.

- Client port

You may want to use rules based on the client port if your clients are using some kind of software that uses a specific client port when making requests.

For example, you could create a rule that says that any requests with a client port of 10002 will have to use a set of special fast servers because you know

that any client requests with that port are coming from an elite group of customers.

- Always true

A rule can be created that is *always true*. Such a rule will always be selected, unless all the servers associated with it are down. For this reason, it should ordinarily be at a lower priority than other rules. You can even have multiple always true rules, each with a set of servers associated with it. The first true rule with an available server is chosen.

For example, assume you have six servers. You want two of them to handle your traffic under all circumstances, unless they are both down. If the first two servers are down, you want a second set of servers to handle the traffic. If all four of these servers are down, then you will use the final two servers to handle the traffic. You could set up three always true rules. Then the first set of servers will always be chosen as long as at least one is up. If they are both down, one from the second set will be chosen, and so forth.

You can define more than one always true rule, and thereafter adjust which one gets executed by changing their priority levels.

We recommend that you make a plan of the logic that you want Dispatcher to follow before you start adding rules to your configuration.

4.9.1 How Are Rules Evaluated?

All rules have a name, type, priority, begin range, and end range, and may have a set of servers. Rules are evaluated in priority order, with lower priority rules evaluated first. In other words, a rule with a priority of 1 will be evaluated before a rule with a priority of 2. The first rule that is satisfied will be used. Once a rule has been satisfied, no further rules are evaluated.

For a rule to be satisfied, it must meet two conditions:

1. The predicate of the rule must be true. That is, the value it is evaluating must be between the begin and end ranges. For rules of type `true`, the predicate is always satisfied, regardless of the begin and end ranges.
2. If there are servers associated with the rule, at least one of them must be available to forward packets to.

If a rule has no servers associated with it, the rule only needs to meet the first condition to be satisfied. If no rules are satisfied, a server will be selected from the full set of servers available on the port.

4.10 Rule-Based Load Balancing Scenario

In this section we show you how it is possible to integrate the load balancing rules in a working scenario.

The scenario we built to experiment with rules was more complex than the other scenarios we have seen in this chapter, since it involved not only the Load Balancing component of IBM WebSphere Performance Pack, but also the File Sharing component. Our scenario involved a Web client machine sending requests to a Web site, composed by three clustered Web servers balanced by a Dispatcher machine. The three Web servers shared the same Web content by

using the File Sharing component of IBM WebSphere Performance Pack. One of the three Web servers was an AIX machine running both as File Sharing client and File Sharing server. The two other Web servers ran as File Sharing client only.

4.10.1 Network Environment

The network environment we used in this scenario is described in the following table:

Table 8. Rule-Based Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM PC 365	wtr05193	9.24.104.239	Windows NT Server 4.0	Web Client
IBM RS/6000 43P	1) rs600023 2) clusterend	1. 9.24.104.128 2. 9.24.104.105	AIX 4.3.1	Dispatcher
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3.1	1) File Sharing Server and Client 2) Web Server
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	1) File Sharing Client 2) Web Server
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	1) File Sharing Client 2) Web Server

On the Windows NT Server machines listed in the above table, Service Pack 3 has also been installed.

Notice that all the machines that took part in this scenario were connected to the same token-ring network.

We started from a situation in which we had already specified a running Dispatcher configuration, with a cluster of three Web servers for the HTTP service on port 80, and with the Manager activated. To be sure that the three Web servers distributed the same content, we included them in an AFS environment (see Chapter 2, "File Sharing Component" on page 17).

Notice that the Web server function in the three HTTP servers was provided by Lotus Domino Go Webserver 4.6.2.5. Netscape Navigator 4.6 was running on the Web client machine. The File Sharing client, File Sharing server and Dispatcher functions were provided by IBM WebSphere Performance Pack Version 1.0.

The following figure shows a graphical representation of the environment we used:

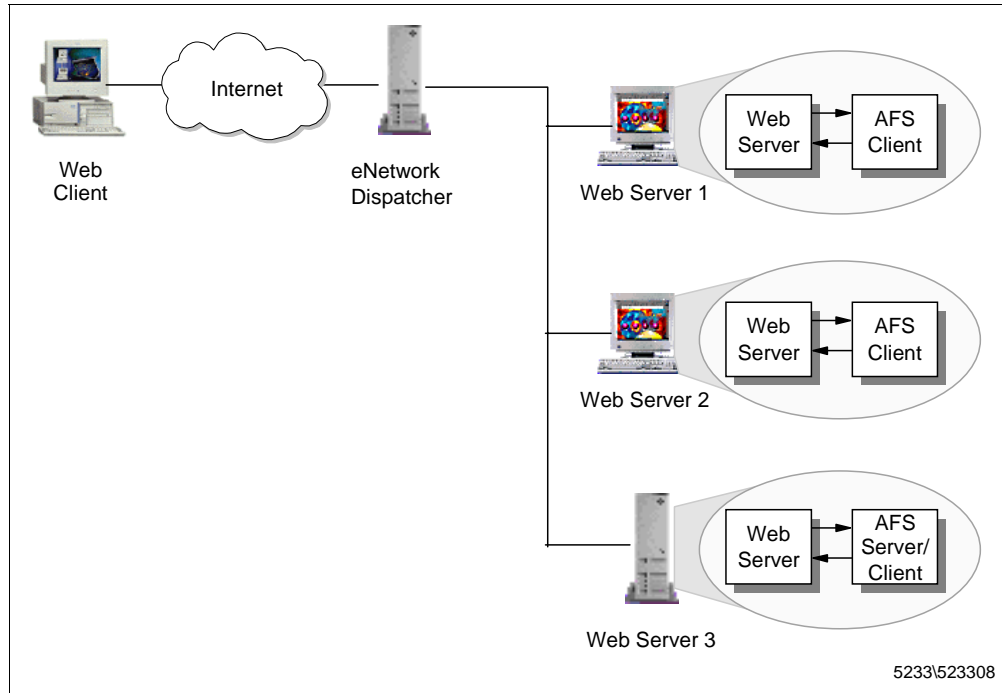


Figure 247. Graphical Representation of the Rule-Based Load Balancing Scenario

4.10.2 Rules Configuration

We added rules to our Dispatcher using the graphical user interface.

In the following figure, a screen capture from the eNetwork Dispatcher GUI is displayed in order to describe the basic running configuration of the Dispatcher that we had set up:

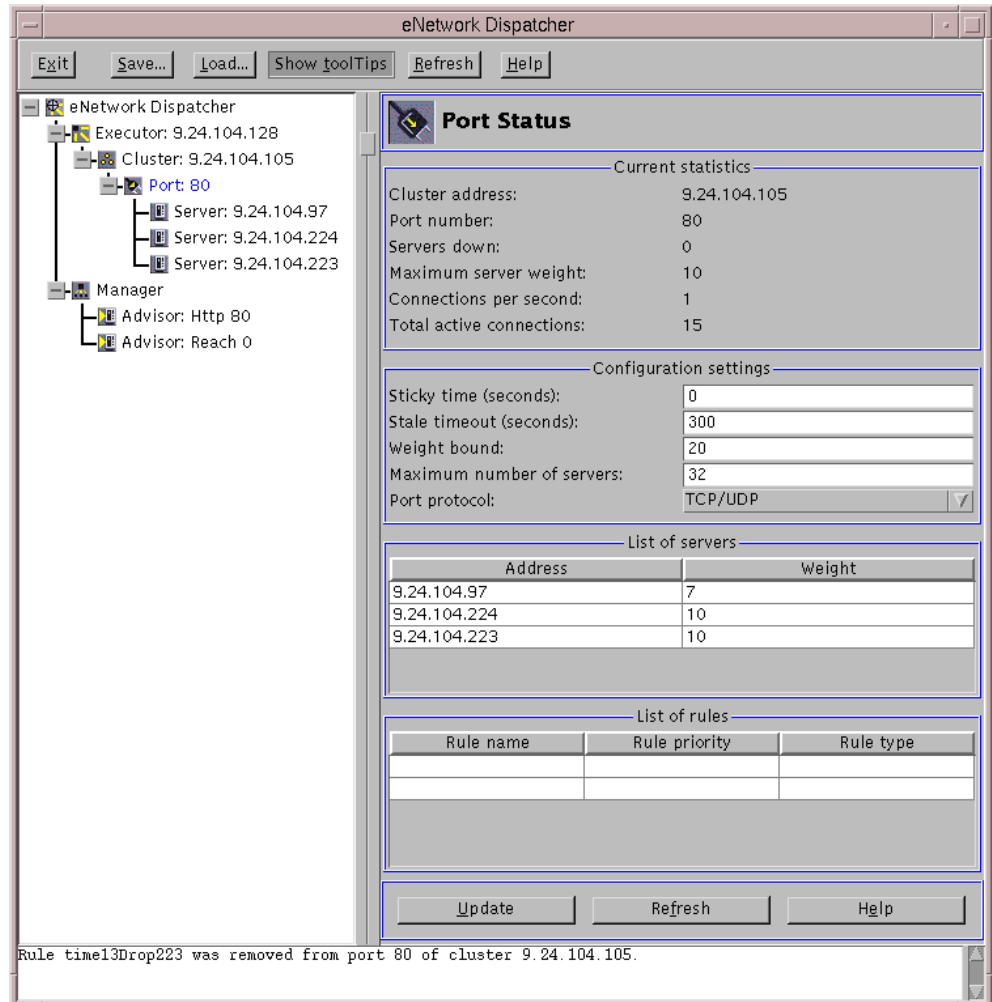


Figure 248. Basic Dispatcher Configuration

To generate the HTTP requests, we deactivated the browser cache on the Web client machine, and set up a direct connection to the Internet without passing through a proxy server.

Before adding a rule, we submitted a set of requests from the Web client and we verified that the Dispatcher was able to select each of the three servers members of the cluster, as shown in the following figure from the Dispatcher GUI Monitor, which shows the current new connections on port 80:

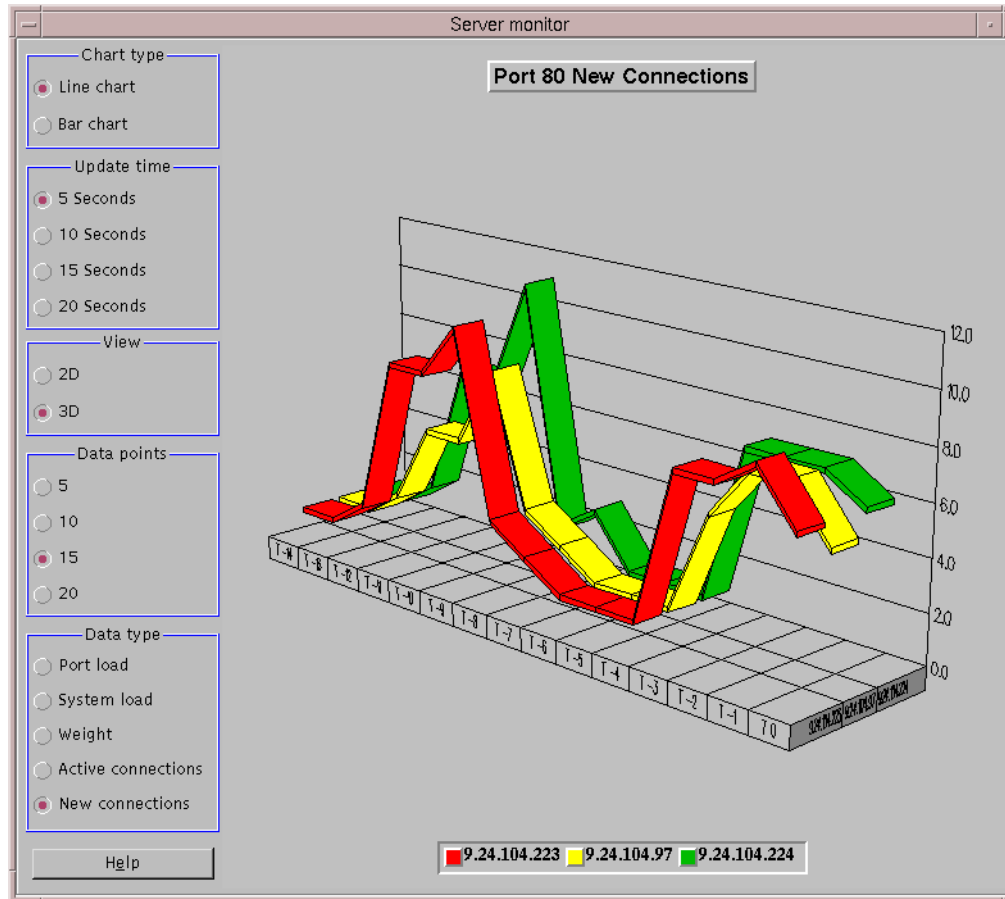


Figure 249. Dispatcher GUI Monitor

At this point, we wanted to add a rule based on the time of day. What we wanted the Dispatcher to do was to distribute the incoming HTTP requests, from 13:00 through 14:59, only to the two Web server machines with IP addresses 9.24.104.97 and 9.24.104.223. Outside the specified time, all the three Web servers should have been active.

To add the rule, we right-clicked on the item **Port 80** in the left panel of the eNetwork Dispatcher GUI, and we got a pop-up menu, as shown in the following figure:

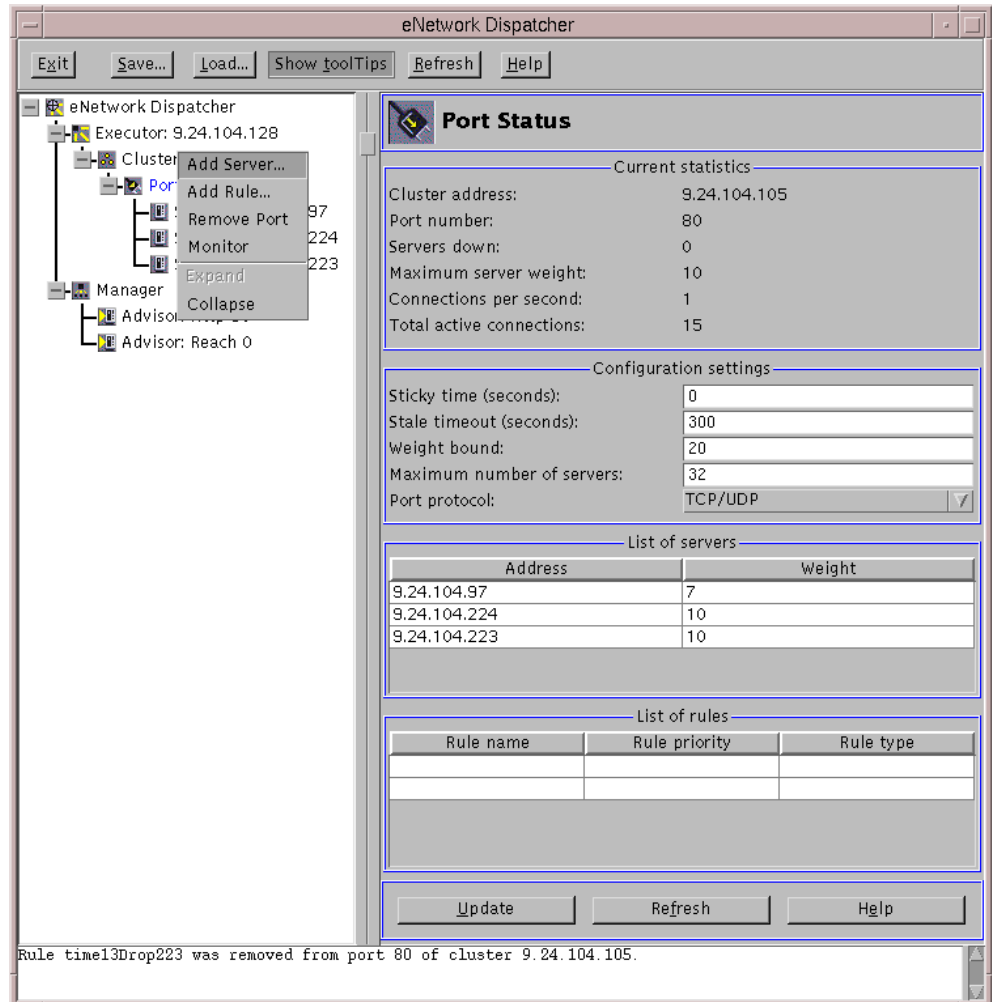


Figure 250. Adding a Rule

Then we selected **Add rule...** and the Add a Rule panel was displayed:

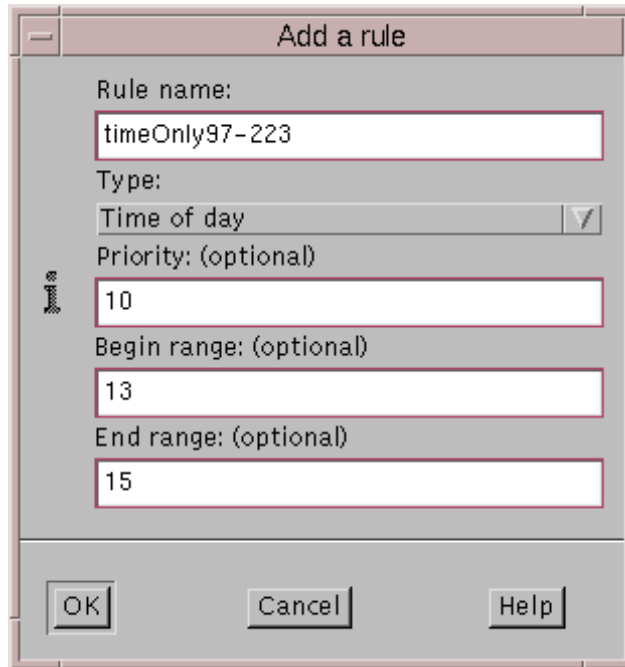


Figure 251. Add a Rule Panel

From this dialog you can specify a new rule with the following parameters:

- You assign a mnemonic name to the rule. In our case, we entered `timeOnly97-223`.
- You then select the type of the rule. In our case, we chose **Time of Day**. Other choices for the rule type would be:
 - `ip`
The rule will be based on the client IP address.
 - `connection`
The rule will be based on the number of connections per second on the port. This rule will work only if the Manager is running.
 - `active`
The rule will be based on the number of active connections total on the port. This rule will work only if the Manager is running.
 - `port`
The rule will be based on the client port.
 - `true`
The rule will always be true.
- Then we specified the priority of the rule. Priorities establish the order in which rules will be reviewed. This parameter accepts integer values, and we set it to 10.

If you do not specify the priority of the first rule you add, Dispatcher will set it by default to 1. When a subsequent rule is added, by default its priority is calculated to be 10 + the current lowest priority of any existing rule. For

example, assume you have an existing rule whose priority is 30. You add a new rule and set its priority at 25 (which, remember, is a higher priority than 30). Then you add a third rule without setting a priority. The priority of the third rule is calculated to be as $30 + 10 = 40$.

In our case, we specified a value of 10 for the priority

- Then we specified the Begin range parameter and the End range parameters.

The Begin range parameter represents the lower value in the range used to determine whether or not the rule is true. The type of values you can assign to it depends on the type of the rule. The type of values and its default are listed here, depending on the rule type:

- ip

The address of the client as either a symbolic name or in dotted-decimal format. The default is 0.0.0.0.

- time

An integer. The default is 0, representing midnight.

- connection

An integer. The default is 0.

- active

An integer. The default is 0.

- port

An integer. The default is 0.

The End range parameter represents the higher value in the range used to determine whether or not the rule is true. The type of values you can assign to it depends on the type of the rule. The type of value and its default are listed here, depending on the rule type:

- ip

The address of the client as either a symbolic name or in dotted-decimal format. The default is 255.255.255.254.

- time

An integer. The default is 24, representing midnight.

Note that when defining the Begin range and End range parameters for time intervals, each value must be an integer representing only the hour portion of the time; portions of an hour are not specified. For this reason, to specify a single hour, say, the hour between 3:00 and 4:00 a.m., you would specify a Begin range of 3 and an End range also of 3. This will signify all the minutes between 3:00 and 3:59. Specifying a Begin range of 3 and an End range of 4 would cover the two-hour period from 3:00 through 4:59.

In our case, we wanted the rule to be true the time from 13:00 through 15:59, and so we set Begin range to 13 and End range to 15.

- connections

An integer. The default is $2^{32} - 1$.

- active

An integer. The default is $2^{32} - 1$.

- port
An integer. The default is 65535.

At this point we chose the **OK** button in Figure 251 on page 308. The GUI reported the configuration was updated with the added rule, as shown in the following figure:

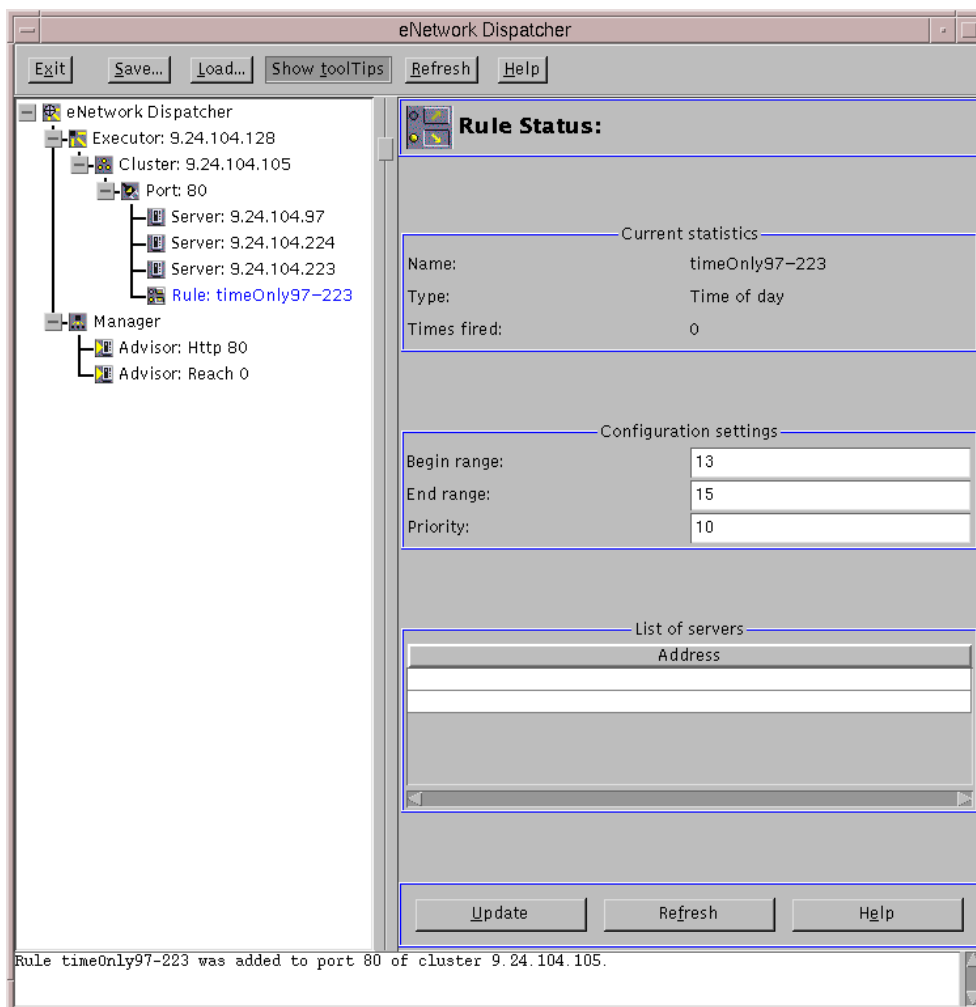


Figure 252. The Configuration Has Been Updated

Once the rule has been added, you need to perform another configuration step, which is to define which servers to use if the rule is true.

As mentioned before, we wanted to use only the Web servers with IP addresses 9.24.104.97 and 9.24.104.224 during the time period from 13:00 through 14:59, and use the three Web servers during the rest of the day.

To add a server to a rule, right-click on the item corresponding to the rule you want to configure on the left side of the eNetwork Dispatcher GUI. You will get a pop-up menu, as shown in the following figure:

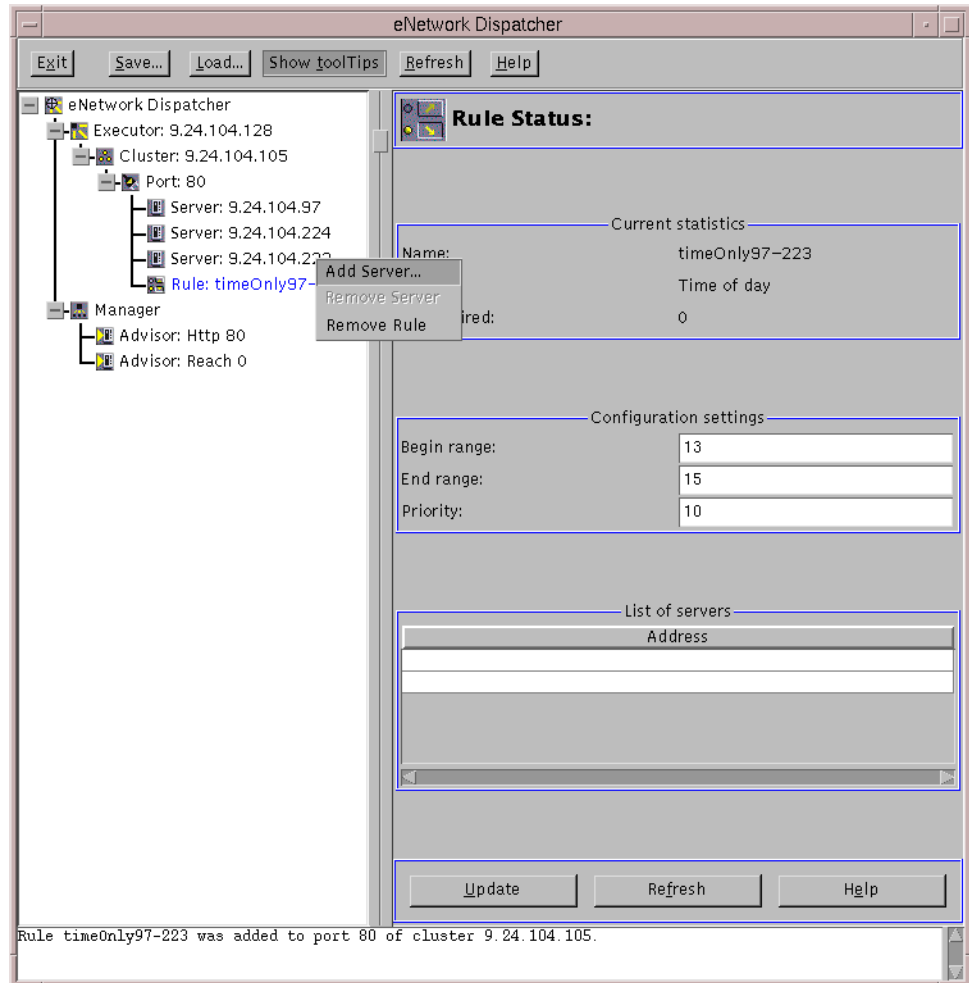


Figure 253. Adding a Server to a Rule

Select **Add Server...** and select the first server you want to add to your rule. Then click **OK**, as shown next:

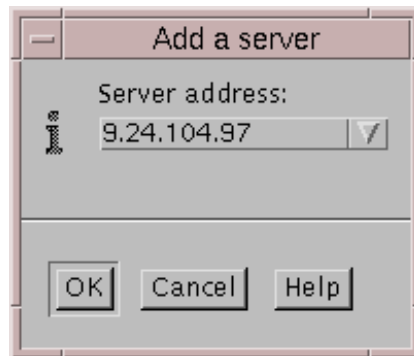


Figure 254. Entering the Server's IP Address

After adding the server with IP address 9.24.104.97, we added in the same way the other one, with IP address 9.24.104.224. The following window shows the updated Rule Status window of our configuration:

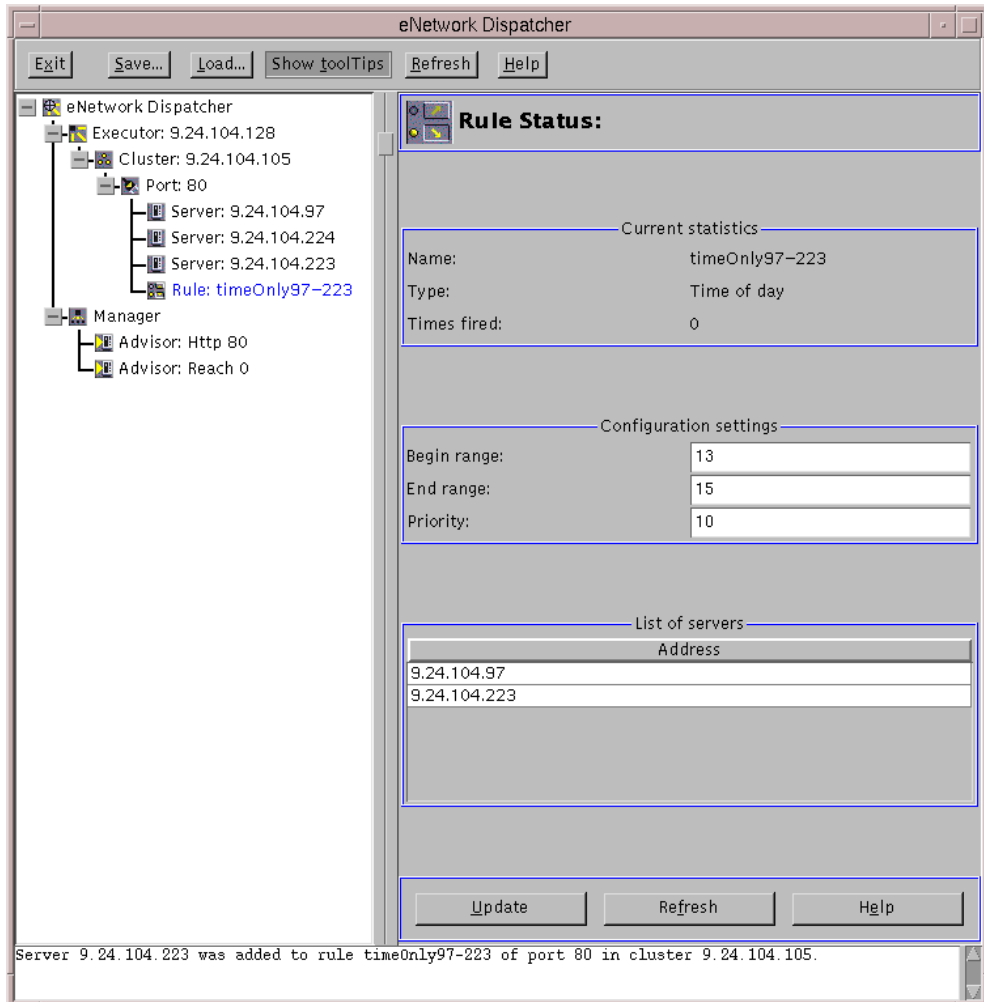


Figure 255. Updated Rule Status Window

This step completed the rule configuration.

To verify that the configuration shown was working correctly, we submitted again a set of requests from the Web browser to the clustered Web site, but this time we did it between 13:00 and 15:59. The following screen shows the Dispatcher Monitor GUI displaying the new online connections on port 80:

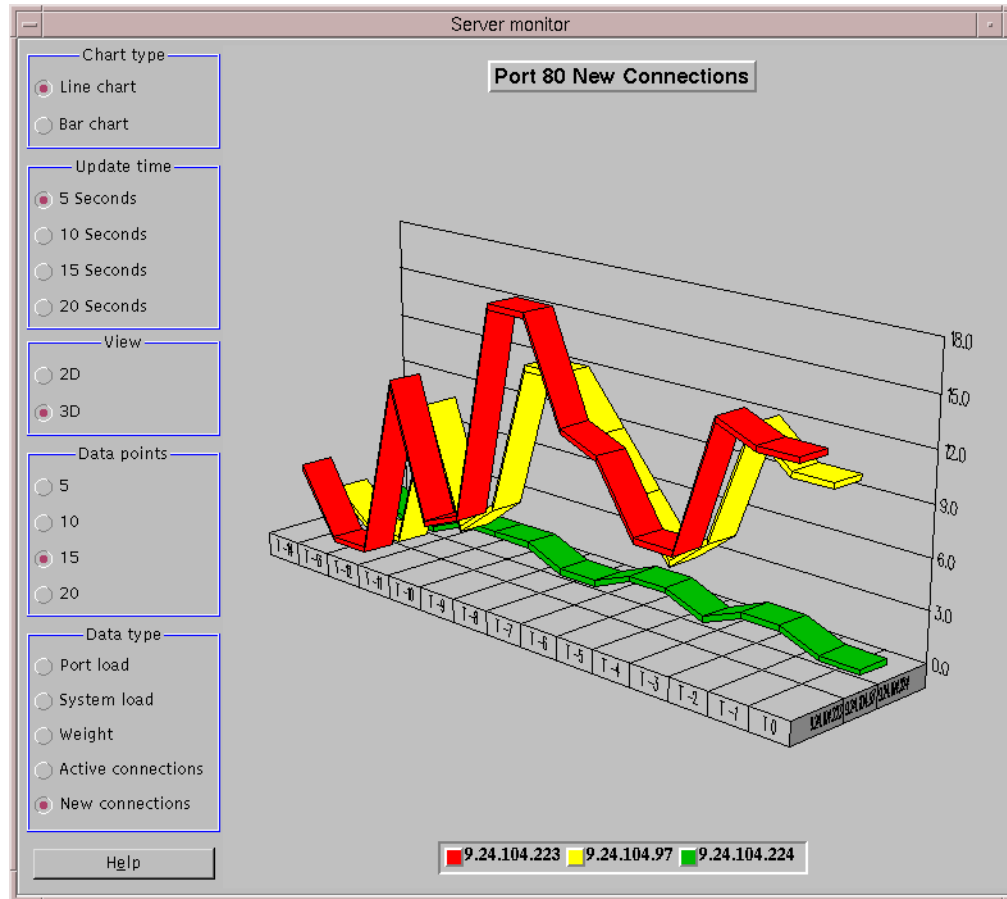


Figure 256. Verifying That the Rule Configuration Has Been Successful

The above figure confirmed that the rule configuration was successful. After 16:00, we generated another HTTP request load from our Web browser. This time, as expected, the Dispatcher time made use of all three Web servers in the cluster.

4.11 Dispatcher High Availability

We have shown in 4.8.1, “ISS Configuration File” on page 295, how to implement ISS high availability.

The Load Balancing component of IBM WebSphere Performance Pack offers built-in high availability also for the Dispatcher function. A standby Dispatcher machine remains ready at all times to take over load balancing should the primary Dispatcher machine fail.

The high availability configuration detects and recovers from network and server failures. The Load Balancing component of IBM WebSphere Performance Pack is smart enough to determine that a server or a network is down. In case of failure, clients lose only the current connections, but they can immediately establish a new connection to the remaining servers with no problems.

The high availability environment involves two Dispatcher machines with connectivity to the same clients, and to the same cluster of servers, as well as

connectivity between the Dispatchers. Both the Dispatchers must be using the same operating systems.

The two Dispatcher machines are usually referred to as *primary* machine and *backup* machine.

The primary machine works normally as a Dispatcher, and stands in the *active* state, balancing the load among the servers of its clusters. The backup machine, configured in a very similar way to the primary machine, stands in *standby* mode unless the primary fails. The two machines are synchronized, and only the primary routes packets, while the backup is continually updated.

The two machines establish communication to monitor the status of each other, referred to as a *heartbeat*, using a port that you can choose. If the primary fails, the backup detects this failure, switches to active state, and begins to take over the routing packets. When the primary machine is operational again, but in standby state, you can either decide that it again *automatically* becomes the active machine, or leave it in standby mode. In this case, you will have to act *manually* if you want it to become the active machine again.

We see in 4.12, “Dispatcher High Availability Scenario” on page 314 how to implement Dispatcher high availability. In 4.13, “Firewall High Availability Using the Load Balancing Component” on page 338 we show how it is possible to use the high availability feature incorporated in the Dispatcher to provide firewall high availability.

4.12 Dispatcher High Availability Scenario

In this section we describe the procedure we followed to configure the Dispatcher high availability feature on our test environment.

4.12.1 Network Architecture

A summary of the hardware, software and network configuration of the environment where we performed our test is reported in the following table:

Table 9. High Availability Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	1) rs600023 2) clusterend	1) 9.24.104.128 2) 9.24.104.105	AIX 4.3.1	Primary Dispatcher
IBM RS/6000 43P	1) august 2) clusterend	1) 9.24.104.46 2) 9.24.104.105	AIX 4.3.1	Backup Dispatcher
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3	Web Server
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	Web Server
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	Web Server

On the two PCs installed with Windows NT Server 4.0, we also applied the Service Pack 3.

The domain name for all the machines was itso.ral.ibm.com.

Dispatcher Operating Systems when Using High Availability

You can see that we installed both the primary and the backup Dispatcher machines on AIX 4.3.1. The most important thing you should remember when you make your plans for Dispatcher high availability is that the two machines implementing the Dispatcher function must run the same operating system. It is not possible to configure the high availability feature if, for example, one Dispatcher machine runs Windows NT and the other AIX or Solaris.

The following figure offers a graphical representation of the Dispatcher high availability scenario that we implemented:

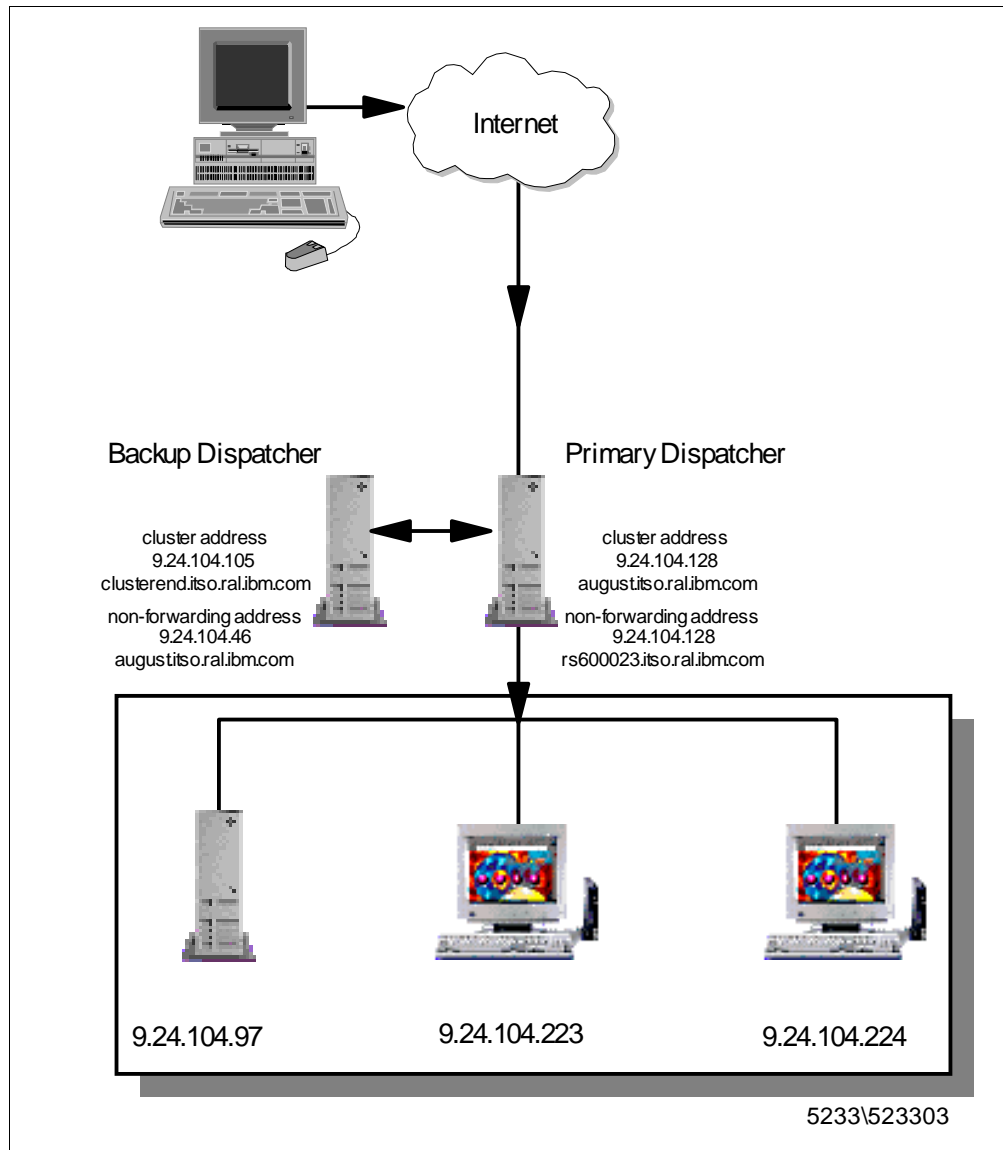


Figure 257. High-Availability Scenario Configuration

4.12.2 Configuration Steps Common on Both the Machines

Before starting the configuration, you need to define the cluster address, which must be the same for both the Dispatcher machines. Both the primary and the backup machines must have their own non-forwarding address for remote administration and configuration.

Note that all the processes (Executor, Manager and Advisors) must run in both machines.

In the following we show the configuration steps for one of the two Dispatcher machines, precisely the one that we elected to use as the backup Dispatcher machine (the machine with hostname august and IP address 9.24.104.46). The same action must be taken on the other machine that will act as the primary Dispatcher machine (the machine with host name rs600023 and IP address 9.24.104.128).

First we started the ndserver component. To do this, from a command line we entered:

```
ndserver start
```

We decided to perform all the configuration activities for the Dispatcher by using the GUI. The same steps could be performed by using the command line, as we see in 4.13, “Firewall High Availability Using the Load Balancing Component” on page 338. We launched the GUI by entering the `ndadmin` command.

Using the GUI, we started the Executor, with 9.24.104.46 as the non-forwarding address; then we defined a cluster for the HTTP service on the TCP port 80. On the port 80 we added the three Web servers. Of course it was necessary to configure the three Web servers to be nodes of the cluster. Then, on the Dispatcher machine, we started the Manager and launched the http Advisor on port 80. We also set the proportions for the Manager and the smoothing index. For details on how to perform all these configuration steps, see 4.6, “Load Balancing Basic Scenario Scenario Using the Dispatcher” on page 246.

In the following windows we show the panels that summarize the configuration.

The first window shows the Executor Status panel for the backup Dispatcher machine. We accepted all the default values.

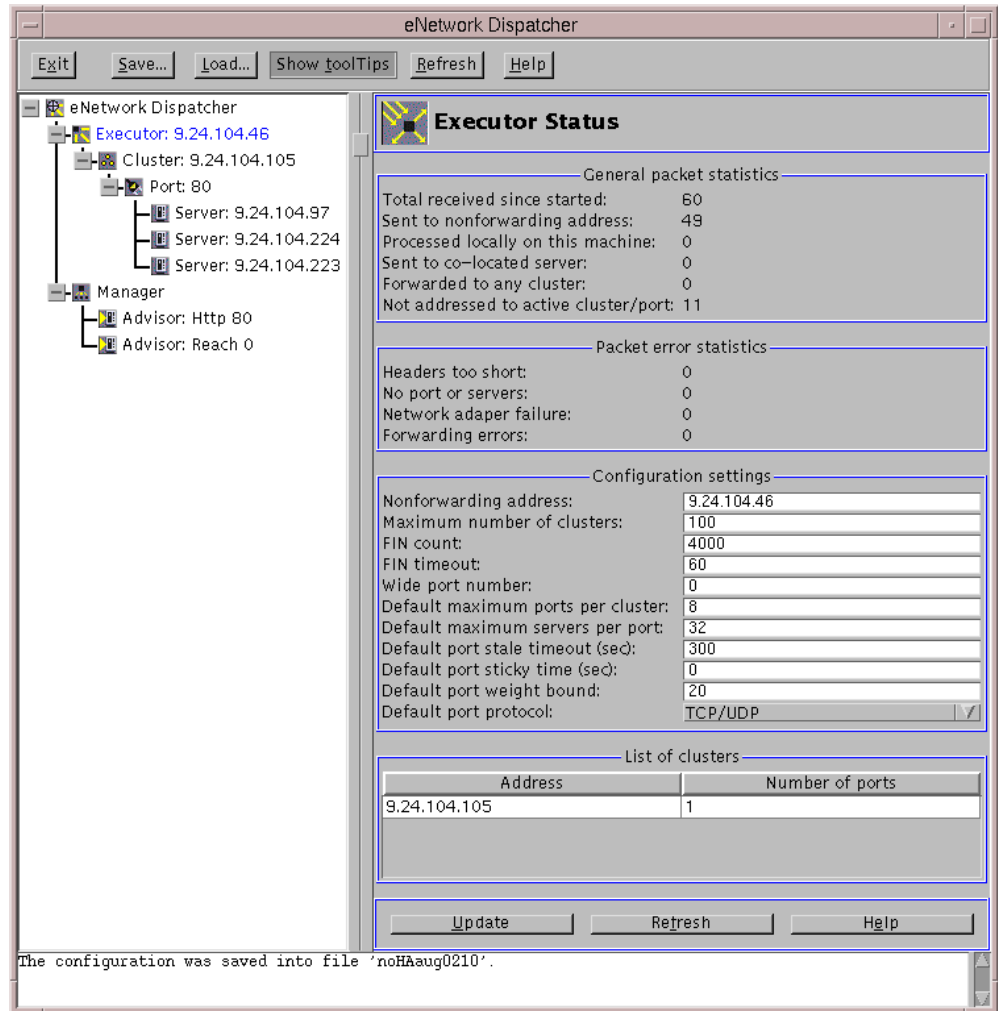


Figure 258. Backup Dispatcher - Executor Status

The Executor Status window for the primary Dispatcher machine was very similar. The only difference was that the non-forwarding address for the primary Dispatcher showed as 9.24.104.105.

The Cluster Status panel is shown in the next figure:

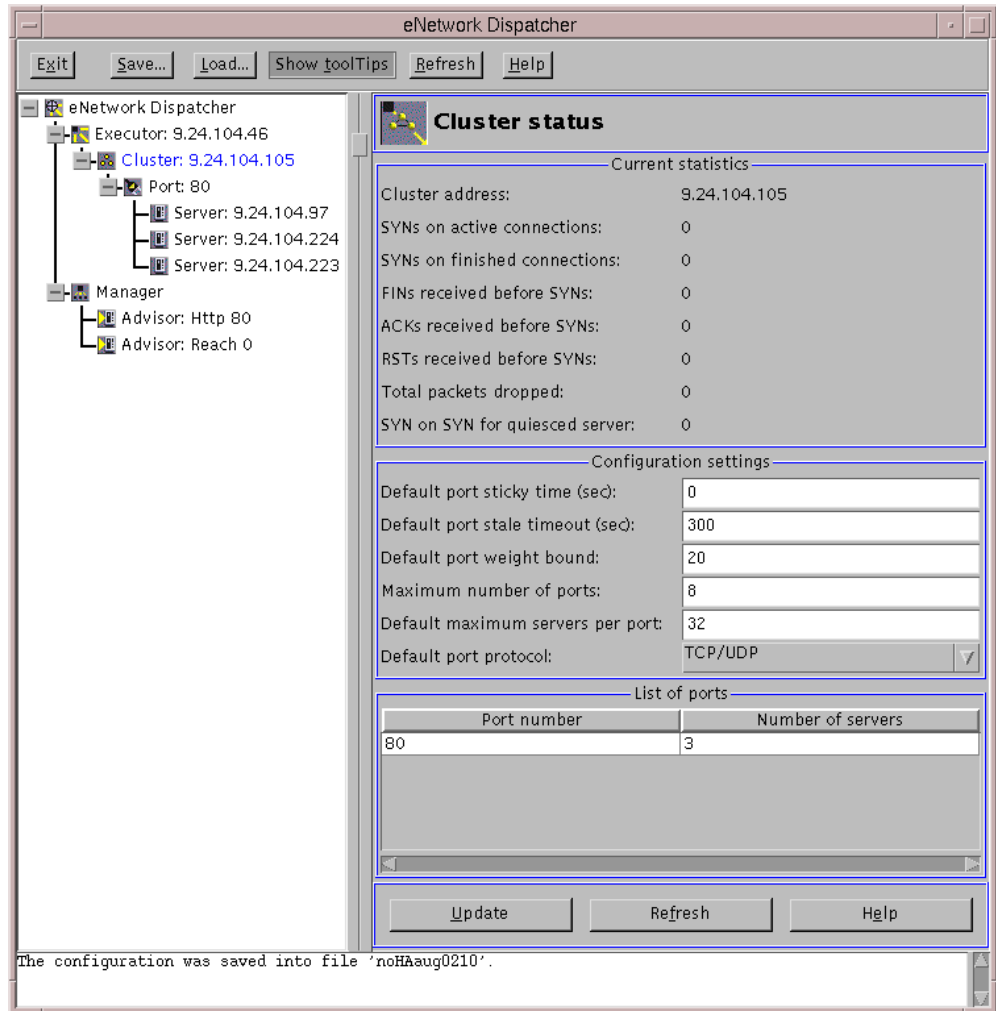


Figure 259. Backup Dispatcher - Cluster Status

In Figure 260 on page 319 you will find the Port Status panel. The addresses of the three Web servers are visible in the figure.

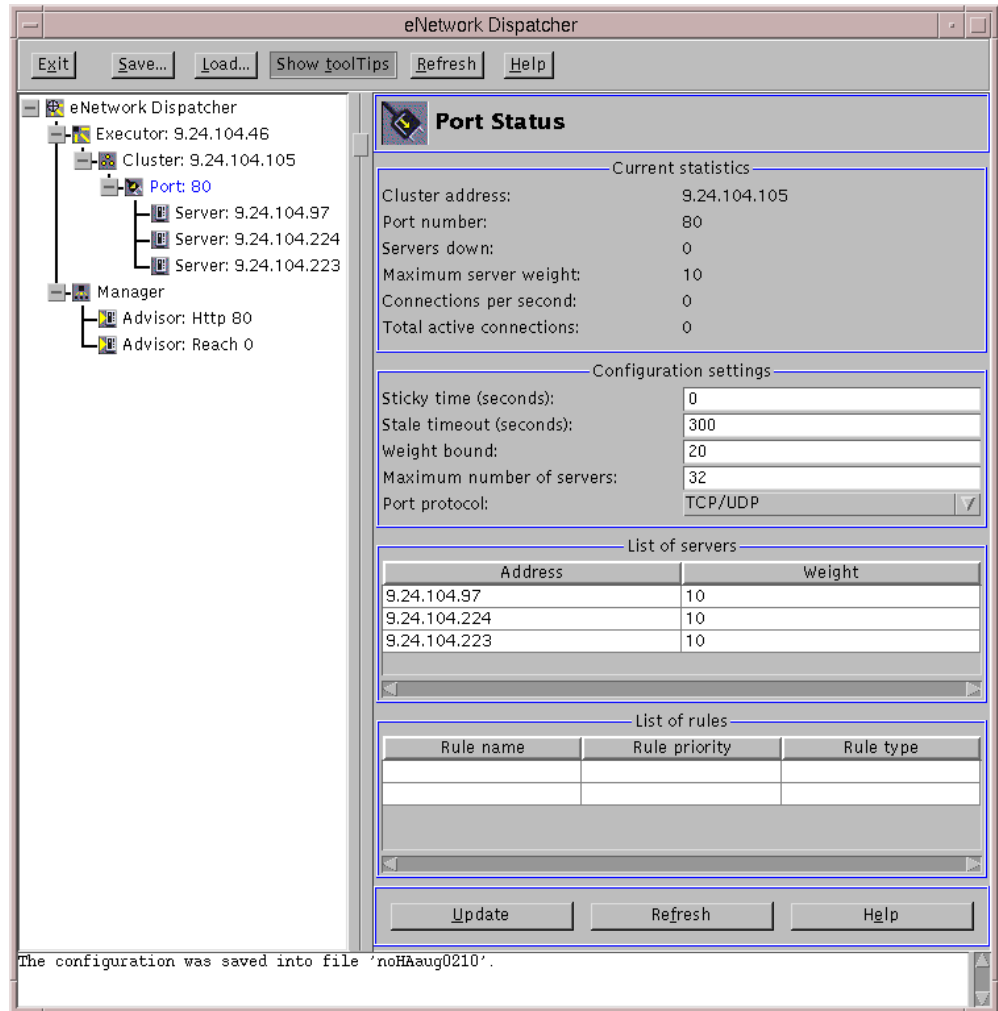


Figure 260. Backup Dispatcher - Port Status

Finally, we show the Manager Status panel. In this case, we did not accept the default values, but we changed the proportions to 49 49 2 0 and the smoothing index values to 6, as shown in the following figure:

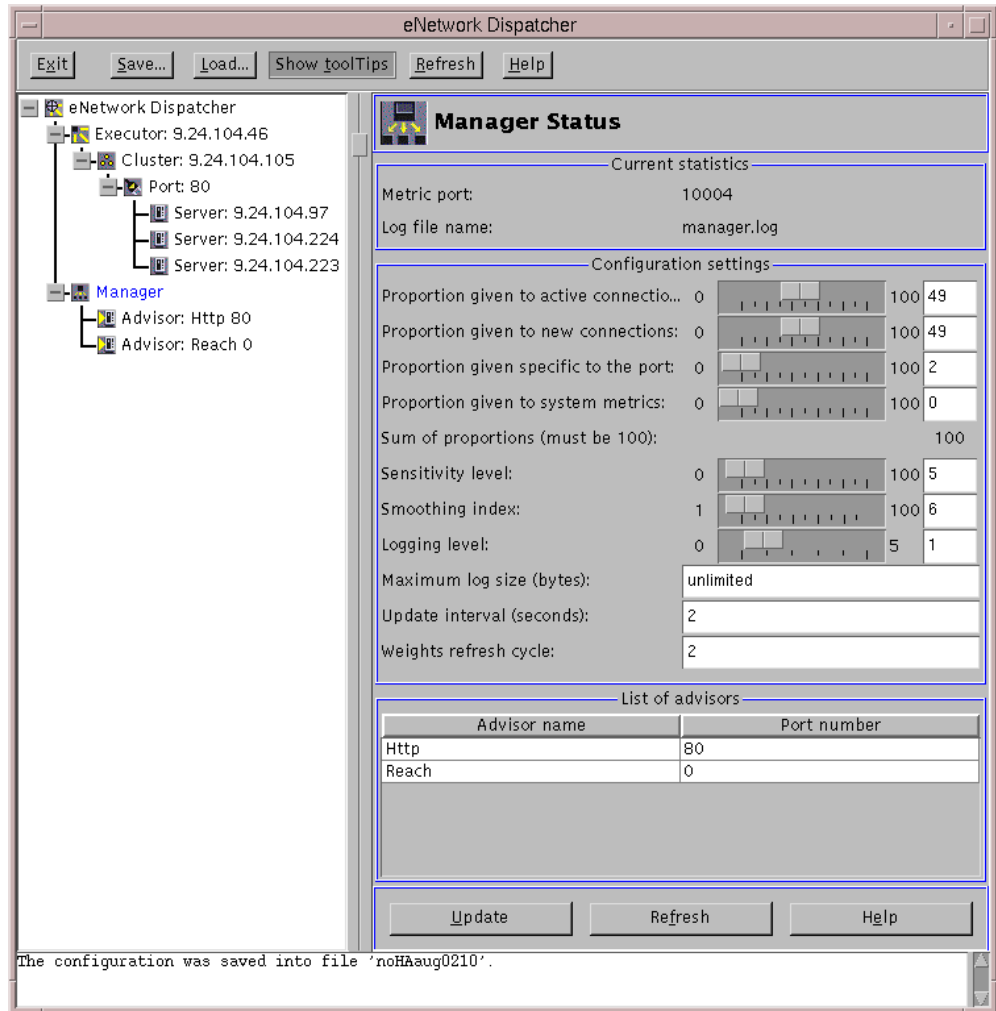


Figure 261. Backup Dispatcher - Manager Status

4.12.3 Configuration Steps for High Availability

Now we start the configuration of the Dispatcher high availability feature. This operation can be done through the GUI configuration panels in a very easy way.

On the backup Dispatcher machine, click on **eNetwork Dispatcher** and from the pop-up menu select **Add Backup**. You will get a dialog similar to the following, which requires you to specify several parameters:

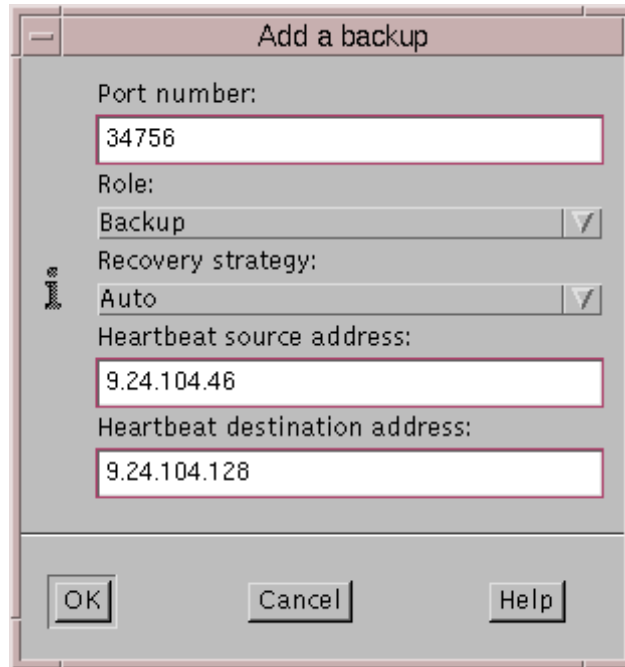


Figure 262. Backup Dispatcher - Add Backup Panel

In the Port number field enter the port over which the two Dispatcher machines will communicate to exchange data for the synchronization of the Dispatcher information. Entering a correct port number is up to you. No default value will appear. We selected a random port number, 34756, but we had to ensure that the selected port was unused on both the Dispatcher machines. To check for that, we issued on both machines the following command:

```
netstat -an | grep 34756
```

We received no message on both machines, which meant that port 34756 was not used by any other process.

We then elected this machine to have the backup role. This can be done by selecting **Backup** from the Role pop-up menu.

We also decided to set the recovery strategy to **Auto**. What is the recovery strategy? If the backup machine at any time detects that the primary machine has failed, it performs a takeover of the primary machine's load balancing functions and becomes the active machine. After the primary machine has once again become operational, it is not said that it becomes active too, but the two machines respond according to how the *recovery strategy* has been configured by the user. There are two kinds of recovery strategy:

1. Automatic

The primary machine resumes routing packets as soon as it becomes operational again.

2. Manual

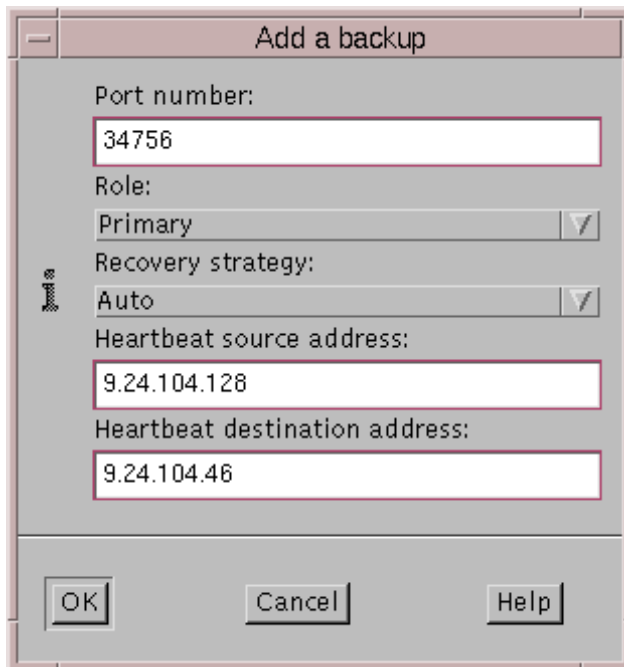
The backup machine continues routing packets even after the primary machine becomes operational. Manual intervention is required to return the primary machine to active state and reset the backup machine to standby.

Notice that the Recovery strategy parameter must be set the same for both machines.

The manual recovery strategy allows you to force the routing of packets to a particular machine, using the `takeover` command, which is also available through the GUI. Manual recovery is useful when maintenance is being performed on the other machine. The automatic recovery strategy is designed for normal unattended operation. We explain to you in more detail the recovery strategy at the end of this section.

Notice that in Figure 262 on page 321 we also added the heartbeat information, the mechanism through which the two Dispatchers continuously detect each other's state.

The following figure shows the values we entered for the other Dispatcher machine; the one we elected as primary Dispatcher:



The image shows a dialog box titled "Add a backup" with the following fields and values:

- Port number: 34756
- Role: Primary
- Recovery strategy: Auto
- Heartbeat source address: 9.24.104.128
- Heartbeat destination address: 9.24.104.46

Buttons at the bottom: OK, Cancel, Help.

Figure 263. Primary Dispatcher - Add Backup Panel

Notice that in this case the heartbeat addresses are inverted as compared with the backup Dispatcher machine (see Figure 262 on page 321). The Role field was set to **Primary** and the port number was the same we used on the backup Dispatcher, since we had already verified that it was not used by any other process.

The heartbeat is a mechanism of message exchange between the two Dispatchers to detect Dispatcher failure. Besides this mechanism of failure detection, there is another failure detection mechanism named *reachability criteria*. When you configure the Dispatcher, you provide a list of hosts that each of the Dispatchers should be able to reach in order to work correctly. You should use at least one host for each subnet your Dispatcher machine uses. The hosts could be routers, IP servers or other types of hosts. Host reachability is obtained by the reach Advisor, which pings the host. Switchover takes place either if the

heartbeat messages cannot go through or if the reachability criteria are met better by the standby Dispatcher than by the primary Dispatcher. To make the decision based on all the available information, the active Dispatcher sends the standby Dispatcher its reachability capabilities. The standby Dispatcher then compares those capabilities with its own and decides whether to switch.

We added the information about the reachability criteria in both the Dispatcher machines in the same way. We right-clicked on the **High Availability** item in the left panel, then selected **Add Reach Target** from the pop-up menu. We got a dialog and there we entered 9.24.104.1, which was the IP address of the gateway machine in our intranet, as shown in the following figure:

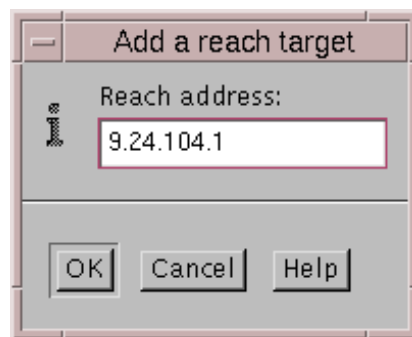


Figure 264. Primary and Backup Dispatcher - Add a Reach Target

The following window shows the High Availability Status panel for the backup Dispatcher:

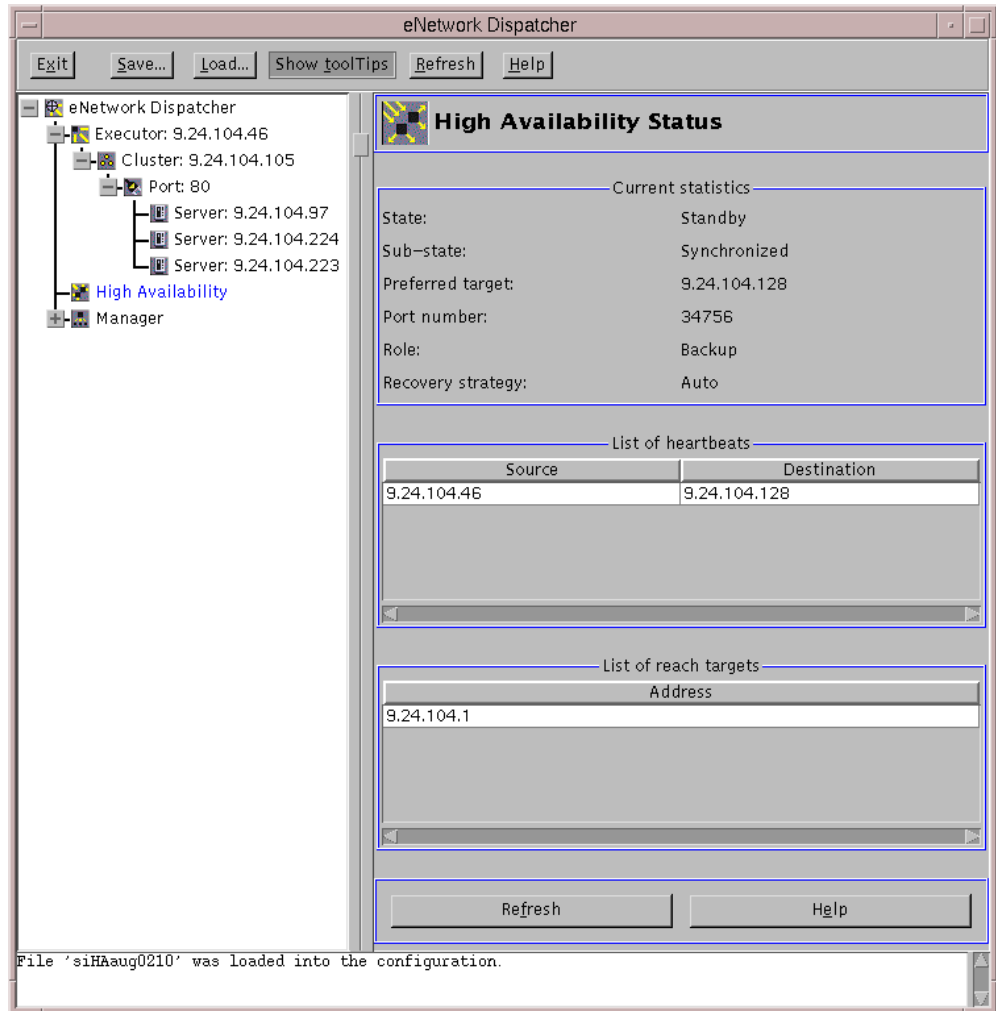


Figure 265. Backup Dispatcher - High Availability Status

In the Current statistics section of the High Availability Status panel, you can see that this machine (the one with backup role) is in standby state and is synchronized with the other machine (the one having primary role). This means that the heartbeat mechanism is working correctly.

The following figure illustrates the High Availability Status panel for the primary Dispatcher, which is in active state and is synchronized with the standby Dispatcher as well:

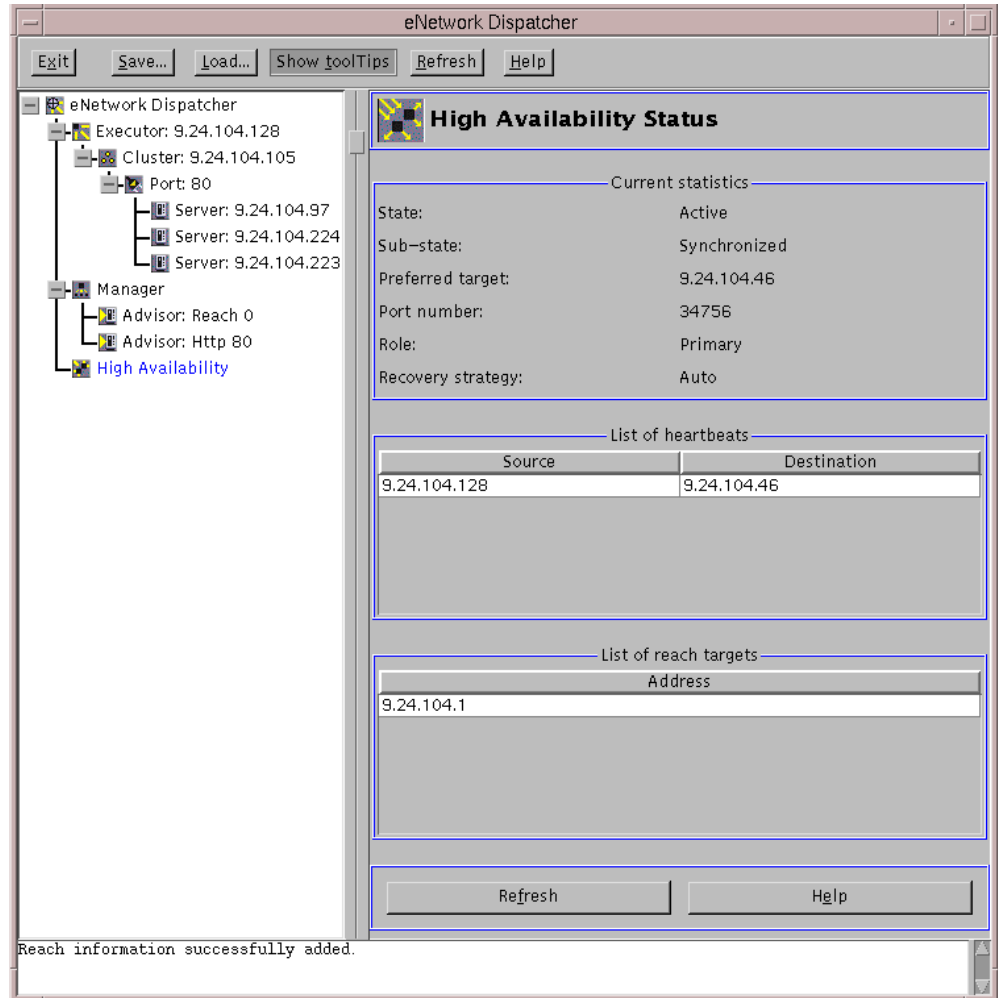


Figure 266. Primary Dispatcher - High Availability Status

To further check whether the heartbeat mechanism was on, from a command line we entered the following command:

```
tcpdump -I -nt -i tr0 port 34756
```

We got the output shown in the following window:

The following figure shows the resulting output displayed on the primary Dispatcher machine:

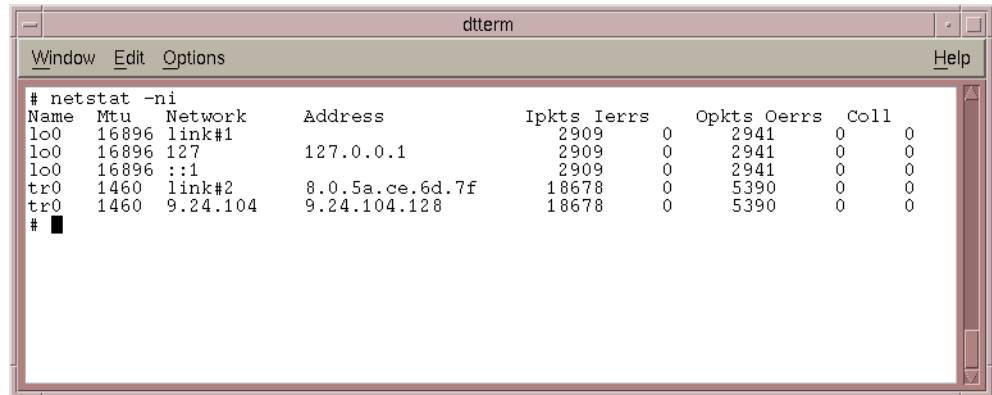


Figure 269. Network Interface Status - Primary Machine

To put the Dispatcher in condition to work, you must configure the alias and loopback devices on both the Dispatcher machines. Precisely, in a high availability configuration:

- On the machine currently in active state, you must alias the cluster address (or each cluster address, if you defined more than one single cluster) to a network interface card.
- On the machine currently in standby state, you must alias the cluster address, (or each cluster address, if you defined more than one single cluster) to the loopback device lo0.
- On any Dispatcher machine (whatever its own current state), you must remove all defined aliases (both on network interface card and loopback device) when the Executor is stopped or before it is started for the first time. In this way no conflicts are possible.

The above-mentioned actions must be issued automatically on each Dispatcher machine when it changes its own state. To ensure that this happens correctly, you need to use three script files, named goActive, goStandby and goInOp, that must be located into a specific directory: /usr/lpp/eND/dispatcher/bin on AIX, and %IBMNDPATH%\Bin\ on Windows NT, where IBMNDPATH is the system environment variable whose value is the path in which the Dispatcher files are installed. Notice that this variable is automatically created during the installation of the Load Balancing component of IBM WebSphere Performance Pack. On our Windows NT systems, the value of IBMNDPATH was D:\Program Files\eND\dispatcher.

Sample versions of the script files are shipped with the product. If you installed the US English product version, they are by default located in the directory /usr/lpp/eND/dispatcher/bin/samples/en_US on the AIX platform and %IBMNDPATH%\Bin\samples\en_US on Windows NT.

A description of the three script files follows, along with the script file we used in our configuration, which was AIX-based. The files used on Windows NT would be pretty similar.

- goActive script

This script deletes loopback aliases and adds device aliases. It will be executed when a Dispatcher machine, planned to be the primary machine in a high availability configuration, goes into active state and begins routing packets.

```
#!/bin/ksh
#
#goActive script
#
#This script is executed when a Network Dispatcher
#goes into the 'active' state and begins
#routing packets.
#
# This script must be placed in the /usr/lpp/eND/dispatcher/bin
# directory in order to be executed
#
CLUSTER=9.24.104.105
INTERFACE=tr0
NETMASK=255.255.255.0
#
echo "Deleting loopback aliases"
ifconfig lo0 delete $CLUSTER
#
echo "Adding device aliases"
ifconfig $INTERFACE alias $CLUSTER netmask $NETMASK
```

Figure 270. goActive Script

- goStandby script

This script deletes every device alias and adds loopback aliases. It will be executed when a Dispatcher machine, planned to be the backup machine in a high availability configuration, goes into standby state without routing packets.

```

#!/bin/ksh
#
#goStandby script
#
#This script is executed when a Network Dispatcher
#goes into the 'standby' state,
#monitoring the health of the active machine,
#but routing no packets.
#
# This script must be placed in the /usr/lpp/eND/dispatcher/bin
# directory in order to be executed
#
CLUSTER=9.24.104.105
INTERFACE=tr0
NETMASK=255.255.255.0
#
echo "Deleting the device aliases"
ifconfig $INTERFACE delete $CLUSTER
#
echo "Adding loopback aliases"
ifconfig lo0 alias $CLUSTER netmask $NETMASK

```

Figure 271. goStandby Script

- goInOp script

This script deletes all device and loopback aliases. This script is executed when a Dispatcher Executor is stopped and before it is started for the first time.

```

#!/bin/ksh
#
#goInOp script
#
#This script is executed when a Network Dispatcher
#executor is stopped (and before the executor is
#initially started).
#
# This script must be placed in the /usr/lpp/eND/dispatcher/bin
# directory in order to be executed
#
CLUSTER=9.24.104.105
INTERFACE=tr0
#
echo "Removing all loopback aliases"
ifconfig lo0 delete $CLUSTER
#
echo "Removing all device aliases"
ifconfig $INTERFACE delete $CLUSTER

```

Figure 272. goInOp script

We copied those three files in the right location, on both the Dispatcher machines, as indicated previously. However, just copying those three files in the right location was not enough to have the two Dispatcher machines be aware of the exact configuration and work in high availability. We needed to make the Dispatchers change their state in order to establish the correct configuration for the aliases on the two machines. In other words, it was necessary to stop and re-start the Executor on both the machines.

First of all, we suggest you to save the current configuration. You can do this by clicking on the **Save...** button on the top of the eNetwork Dispatcher GUI. When we did this on the backup Dispatcher machine, we got the following panel, where we typed `augHA-bck-auto` as the name that we wanted to save the configuration file as:

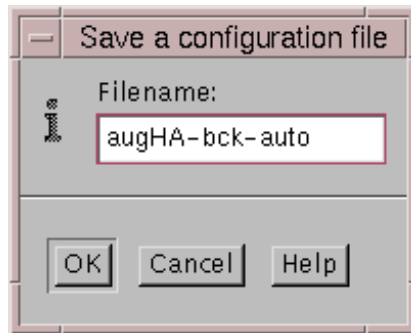


Figure 273. Save Configuration File - Backup Machine

The name we entered for the configuration file in the primary Dispatcher machine was `r23-HA-pri-auto`.

Then we stopped the Executor on both the machines. To do that, right-click on the **Executor** item on the left window of the GUI, then click on **Stop Executor** (see Figure 203 on page 259). At this point, the `golnOp` script has been executed on both machines, but without real effects, since we did not previously define any aliases.

Then we restarted the Executor on both machines. To do so, click the **Load...** button from the top tool bar, choose the configuration file name from the dialog that appears and click **OK**. This is what we did on our backup Dispatcher machine:

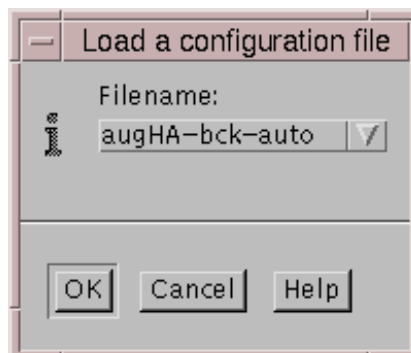


Figure 274. Load Configuration File - Backup Machine

On the primary Dispatcher machine, we loaded the configuration file r23-HA-pri-auto in a very similar way.

After restarting the Executors on both the machines, the primary Dispatcher machine switched its State field to read *Active* (and the *goActive* script was executed on that machine) while the State field in the backup Dispatcher machine changed to *Standby* (and the *goStandby* script was executed on that machine). Moreover both the machines switched their Sub-state to *Synchronized*. Notice that after restarting the Executors, it might be necessary to click the **Refresh** buttons on the eNetwork Dispatcher GUI windows to see the State and the Sub-state fields updated.

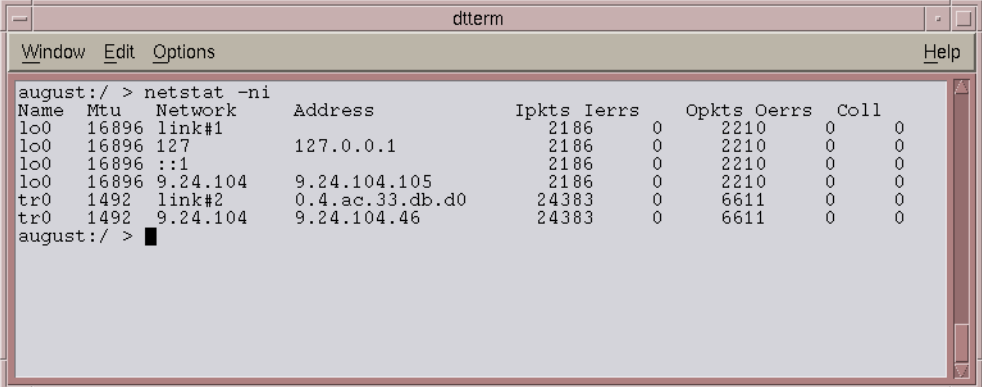
Executing the Scripts

Notice that it would not be really necessary to restart the Executor on the primary and backup machines. The scripts are executable, so you should be able to just run the *goActive* script on the primary machine and the *goStandby* script on the backup machine, and the network interface cards and loopback devices would be configured accordingly.

We wanted to check whether the aliases were correctly added, so from a command line we issued the following command on both the machines:

```
netstat -ni
```

We got the following output on the backup Dispatcher machine, which was in standby state:



```
dtterm
Window Edit Options Help
august:/ > netstat -ni
Name Mtu Network Address Ipkts Ierrs Opkts Oerrs Coll
lo0 16896 link#1 2186 0 2210 0 0
lo0 16896 127 2186 0 2210 0 0
lo0 16896 ::1 2186 0 2210 0 0
lo0 16896 9.24.104 9.24.104.105 2186 0 2210 0 0
tr0 1492 link#2 0.4.ac.33.db.d0 24383 0 6611 0 0
tr0 1492 9.24.104 9.24.104.46 24383 0 6611 0 0
august:/ >
```

Figure 275. Network Interface Status - Backup Machine

This output confirmed that the alias to the cluster address 9.24.104.105 was added on the loopback device lo0.

We got the following output on the primary Dispatcher machine, whose state was marked as active:

```
dtterm
Window Edit Options Help
# netstat -ni
Name Mtu Network Address Ipkts Ierrs Opkts Oerrs Coll
lo0 16896 link#1 127.0.0.1 3130 0 3187 0 0
lo0 16896 127 127.0.0.1 3130 0 3187 0 0
lo0 16896 ::1 3130 0 3187 0 0
tr0 1460 link#2 8.0.5a.ce.6d.7f 25622 0 8012 0 0
tr0 1460 9.24.104 9.24.104.128 25622 0 8012 0 0
tr0 1460 9.24.104 9.24.104.105 25622 0 8012 0 0
# █
```

Figure 276. Network Interface Status - Primary Machine

This output testified that the alias to the cluster address 9.24.104.105 has been added on the network adapter interface tr0.

Now all network and loopback interfaces are correctly configured. The primary Dispatcher is able to route IP packets to the servers belonging to the defined cluster and the backup Dispatcher is able to take over in case the primary Dispatcher should have a failure.

4.12.4 Experimenting with High Availability

We wanted to perform a test to demonstrate that the automatic takeover worked correctly. Several tests could be done, such as disconnecting the network interface on the primary machine, shutting down the primary machine or even turning it off.

We selected a solution that would not damage the stability of our test machines. While the primary machine was in active state, on that machine we issued the command:

```
ifconfig tr0 down
```

This command brought the token-ring interface down.

The backup machine detected that the primary machine was no longer available, and changed to active state, as shown in the following figure:

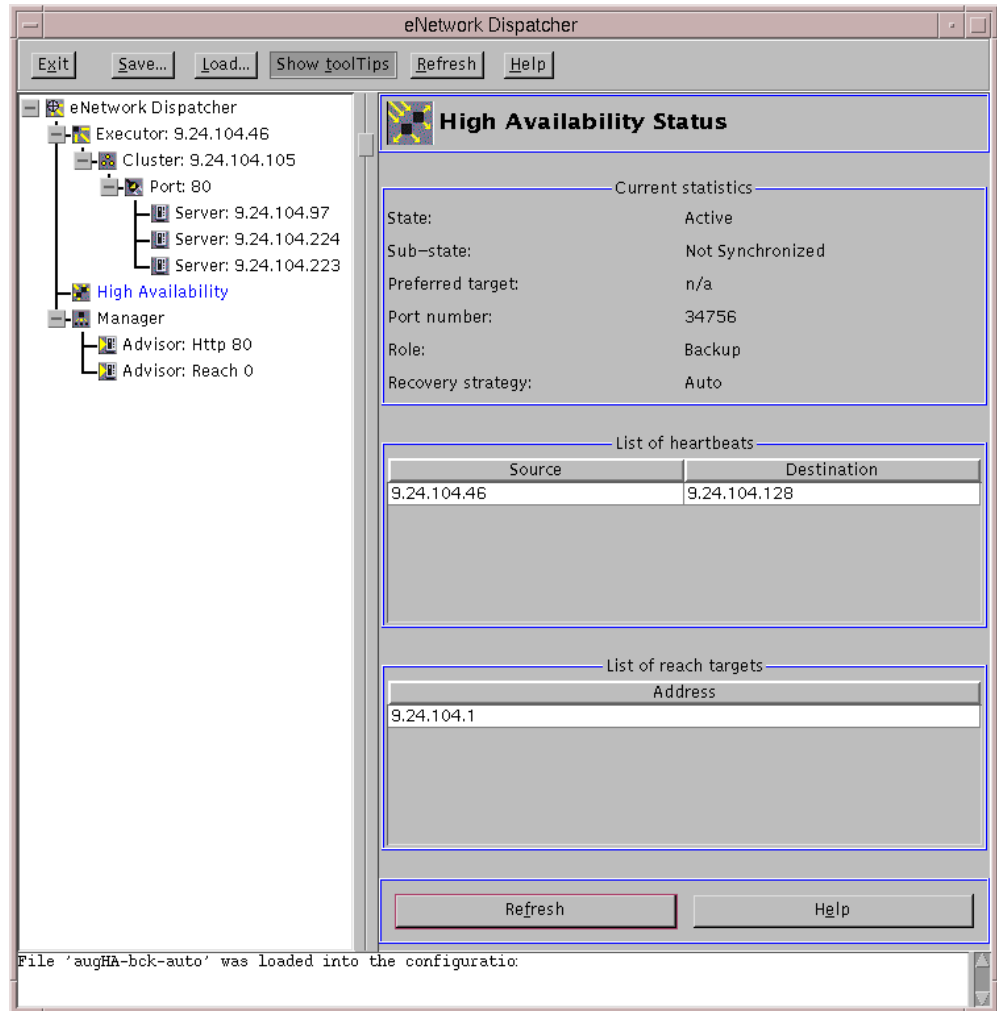


Figure 277. High Availability Status Window

Of course, the Sub-state field appeared as *Not Synchronized*, since the primary machine was no longer able to exchange heartbeat messages.

As soon as the primary machine failed, the `goActive` script file was executed on the backup machine. Then we issued the following command on the primary machine, which became operational again:

```
ifconfig tr0 up
```

Since we set the recovery strategy to `Auto`, the primary machine went immediately to active state, while the backup machine changed its state back to standby, by executing the `goStandby` script.

4.12.4.1 Experimenting with the Recovery Strategy

Now, we want to explain to you in more detail the recovery strategy. Notice that in the configuration we have just described, where we had set the Recovery strategy parameter to **Auto** in both the Dispatcher machines (see Figure 262 on page 321), we found that the `takeover` command had been disabled *on both the machines*. This was visible after right-clicking the **High Availability** item in the

eNetwork Dispatcher GUI. The following screen was captured for example on the backup Dispatcher machine:

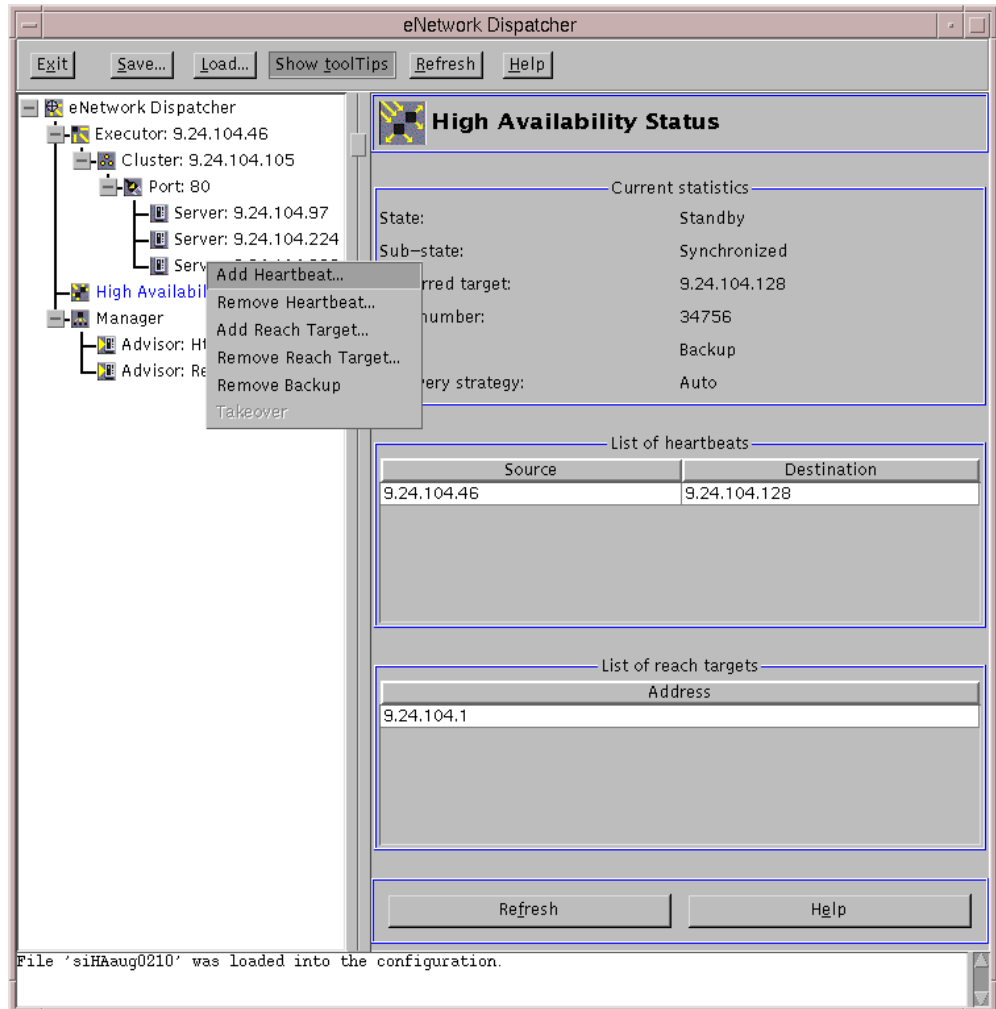


Figure 278. Takeover Disabled When the Recovery Strategy Is Set to Auto

Then we tried another experiment. We defined two new configurations for the two Dispatcher machines, still in high availability mode, but setting the Recovery strategy field to **Manual**. After that we noticed that the Takeover item from the High Availability pop-up menu, corresponding to the `takeover` command, had been enabled on the backup Dispatcher machine, as shown in the following screen:

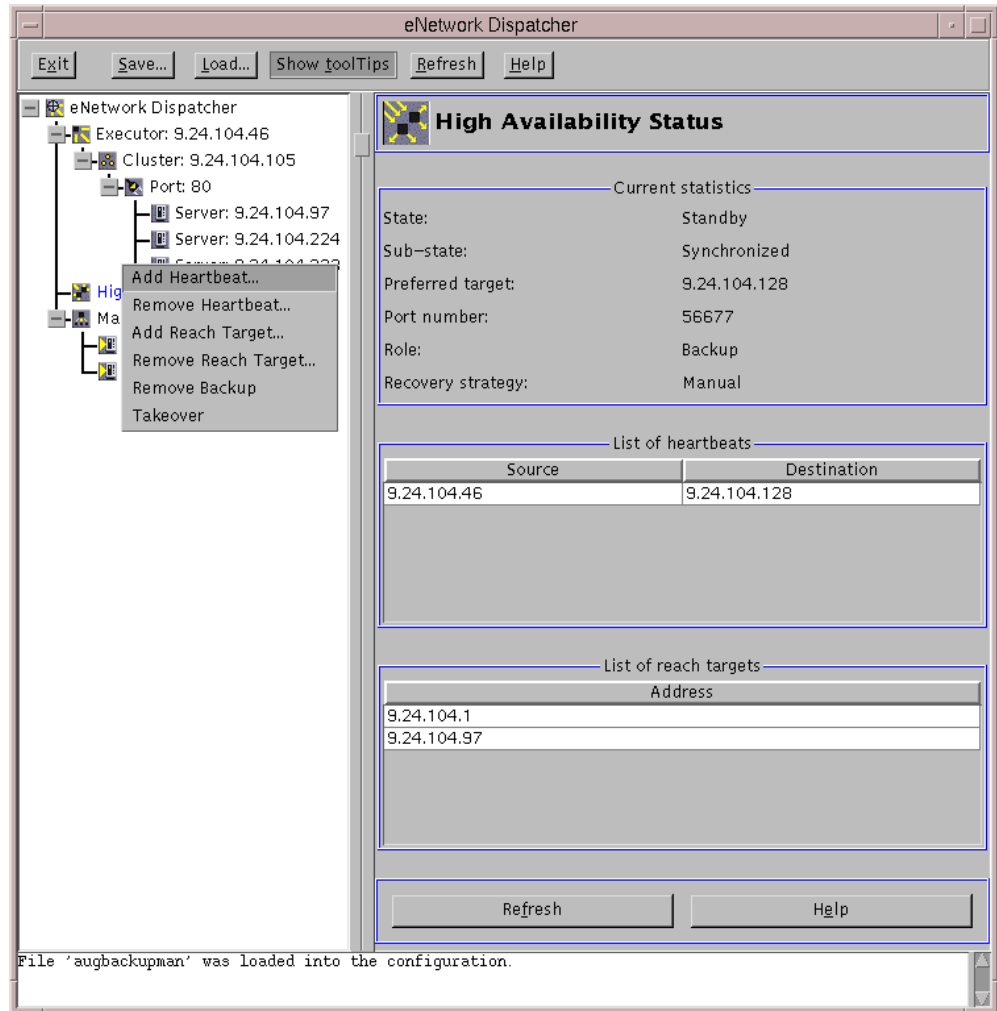


Figure 279. Takeover Enabled When the Recovery Strategy Is Set to Manual

In the above figure you can notice other details:

- We added the IP address 9.24.104.97 of the Web server rs600030 to the list of reach targets.
- We defined another port number, 56677, for the heartbeat mechanism, after verifying that no other processes were using that specific port number.
- We saved the configuration in a new configuration file, which we named augbackupman.

The Takeover item in the High Availability pop-up menu was still disabled in the primary Dispatcher machine, since the state of this Dispatcher machine was already set to Active, as shown in the following figure:

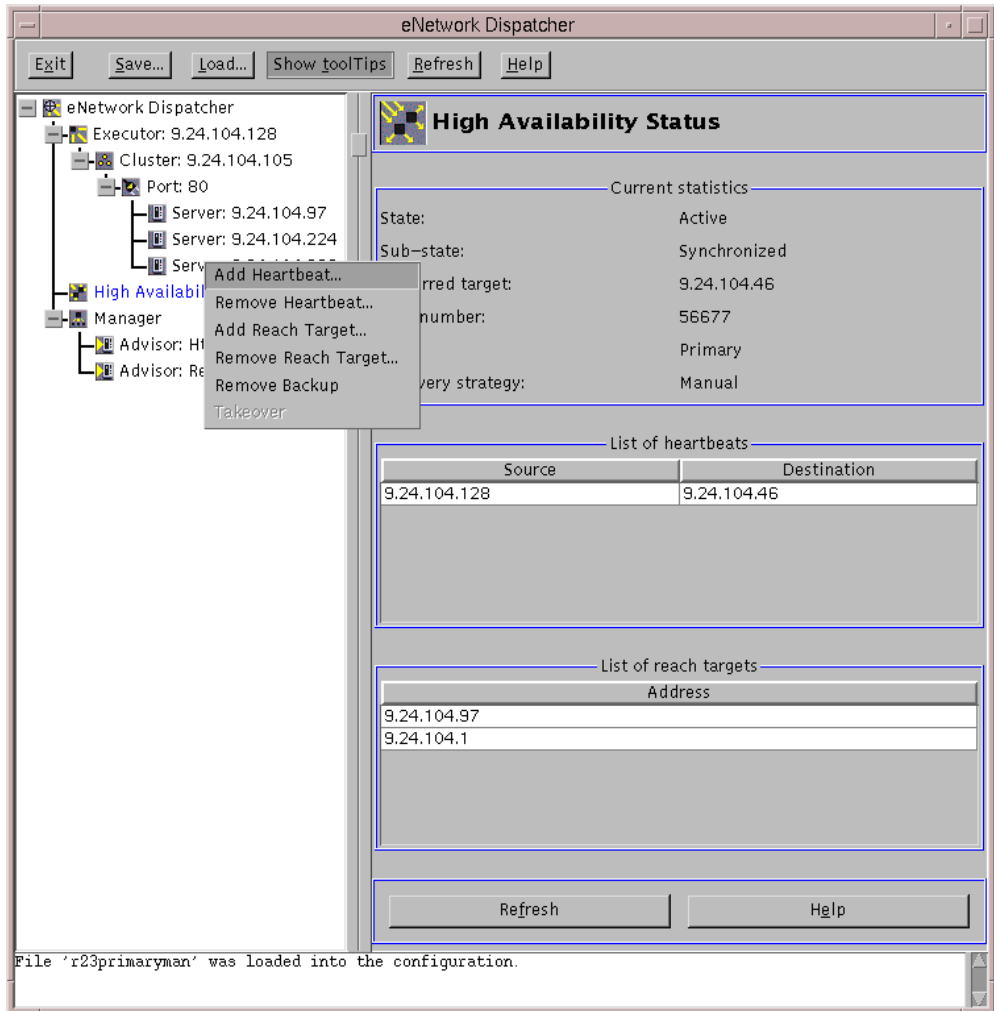


Figure 280. Takeover Disabled on the Primary Dispatcher Machine

Notice that also in the configuration of the primary Dispatcher machine we had made some changes, according to the changes made on the backup machine configuration. You can see that we had saved the new configuration on the primary Dispatcher machine in a new configuration file that we named r23primaryman.

Then, in the eNetwork Dispatcher GUI of the backup Dispatcher machine we selected **Takeover**, as shown in Figure 279 on page 335, and what we got was the following screen:



Figure 281. Takeover Confirmation Window

The reason for this further confirmation window is that the takeover operation is very important in a network environment where two Dispatchers are running in high availability mode. We selected **Yes** and the eNetwork Dispatcher GUI for the backup Dispatcher machine appeared as shown in the following figure:

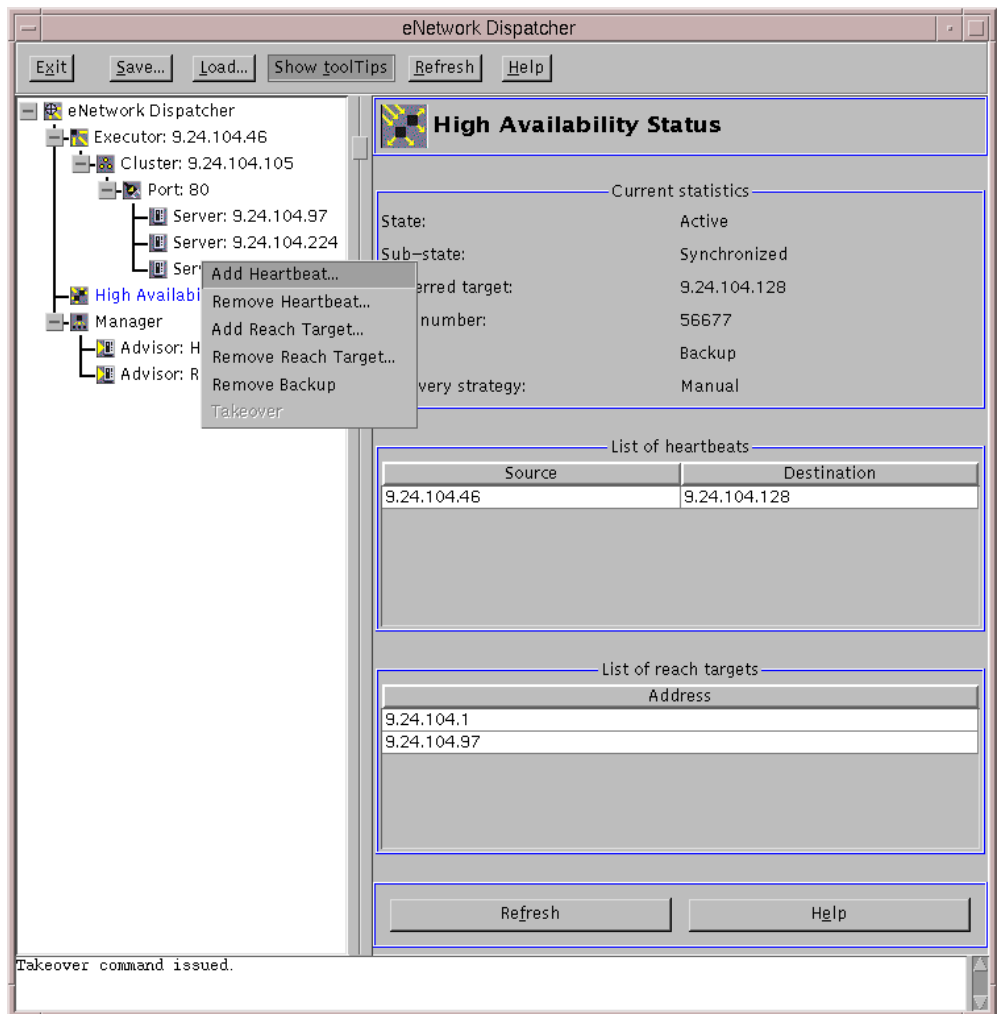


Figure 282. eND GUI on the Backup Machine after the Takeover

Notice that the the Role of that Dispatcher machine remained set to `Backup`, but its state changed to read `Active` and, for this reason, the `Takeover` item was disabled again from the High Availability pop-up menu.

On the primary Dispatcher machine, the Dispatcher state appears set to `Standby`, and the `Takeover` item became enabled, as you can see in the following figure:

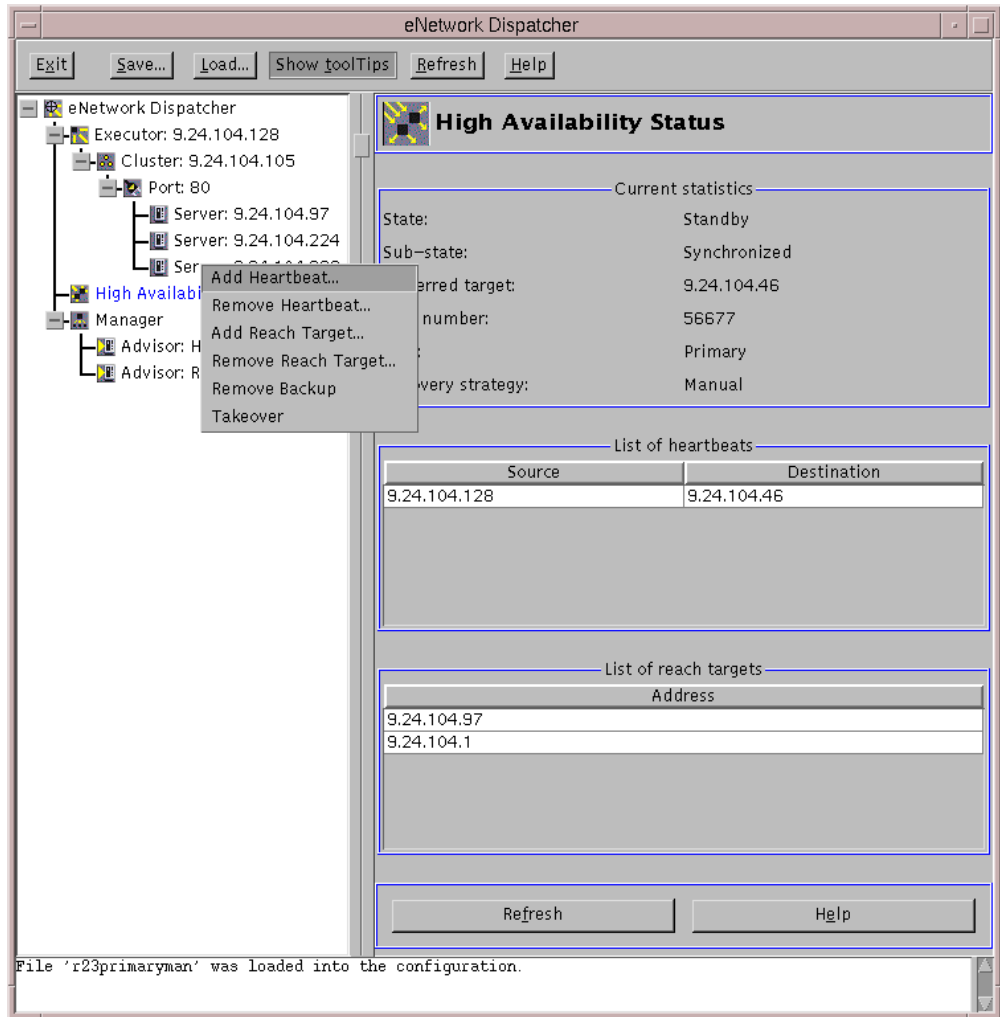


Figure 283. Takeover Enabled on the Primary Dispatcher Machine

This is a confirmation that even if the primary machine is operational when the recovery strategy is set to manual, only a manual intervention can return the primary machine to active state and reset the backup machine to standby. Otherwise the backup machine continues routing packets.

4.13 Firewall High Availability Using the Load Balancing Component

This section contains instructions on how to configure IBM eNetwork Firewall and the Load Balancing Component of IBM WebSphere Performance Pack to provide a highly available firewall configuration. This document does not go into installation instructions for IBM eNetwork Firewall, as these topics are better addressed through the product manual.

The scenario we are going to show now makes use of the Load Balancing component of IBM WebSphere Performance Pack to provide firewall high availability.

4.13.1 Basic Configuration Issues

The Load Balancing component of IBM WebSphere Performance Pack uses cluster addresses to identify the addresses over which it will perform its load balancing. In the scenario we are interested in, the Load Balancing component will not be performing any load balancing, but we still need to use cluster addresses for the highly available portion of the setup. We need to configure two cluster addresses for this scenario: one for the secure interface and one for the non-secure interface. The cluster addresses are the ones that will be transferred from one machine to the other in a fail-over situation. They are also the addresses that clients will point to on the secure and non-secure networks. The configuration of these addresses is done in several scripts that are run by the Dispatcher.

Imagine the following scenario. There are two eNetwork Firewall machines: FW1 and FW2. FW1 will be the primary machine, and FW2 will act as a backup machine. Each machine also has eNetwork Dispatcher installed.

The following figure shows a graphical representation of this scenario:

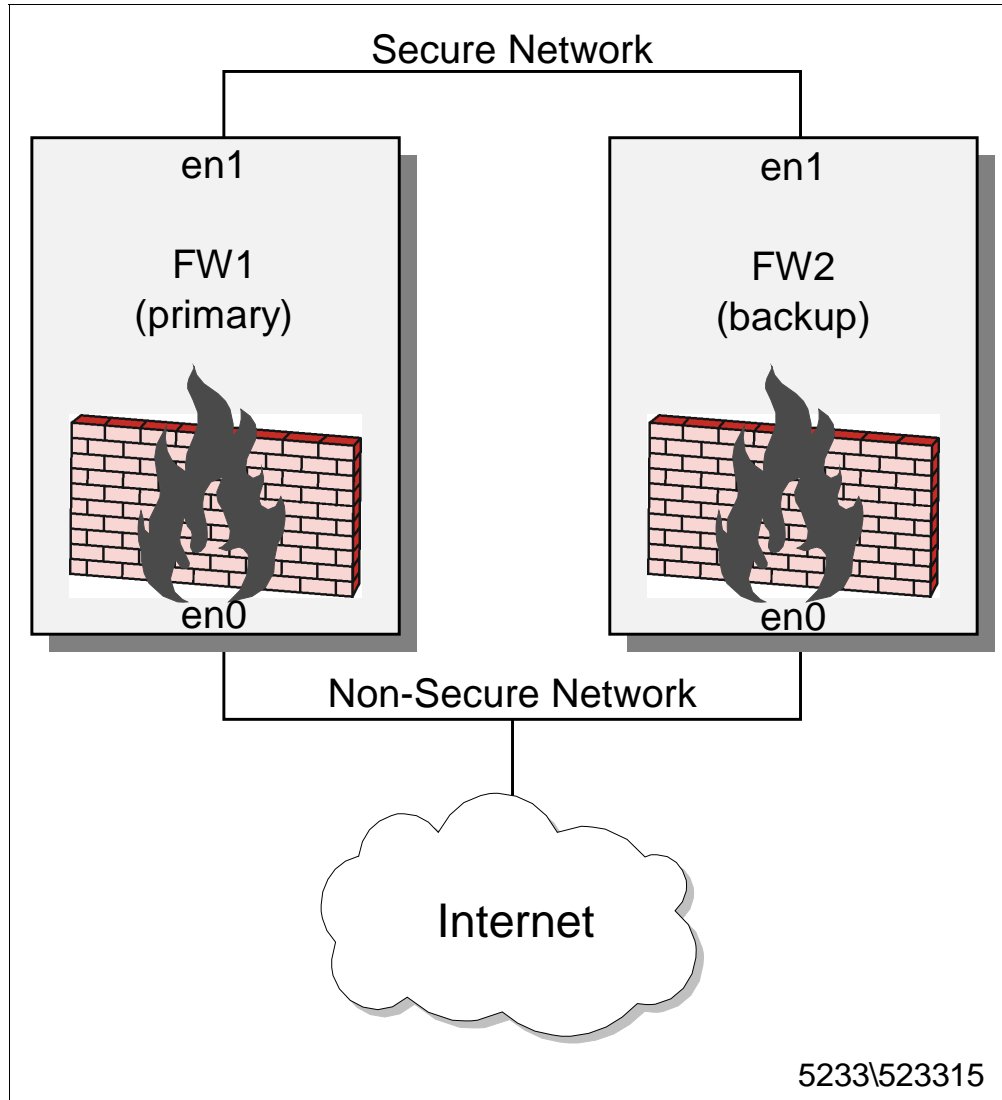


Figure 284. Graphical Representation of the Scenario

For this example, both networks will be using Ethernet. There are three scripts that the Dispatcher uses to fail-over the cluster addresses. They are goActive, goInOp and goStandby. Let's assume both machines are running AIX and are using the following IP address scheme:

Table 10. Basic Configuration

Firewall Machine	FW1	FW2
IP Address for Non-Secure Network	9.67.123.4	9.67.123.5
IP Address for Secure Network	10.0.0.4	10.0.0.5
Cluster Address for Non-Secure Network	9.67.123.8	9.67.123.8
Cluster Address for Secure Network	10.0.0.8	10.0.0.8
Subnet Mask for Non-Secure Network	255.255.248.0	255.255.248.0
Subnet Mask for Secure Network	255.255.255.0	255.255.255.0

Notice that the fact that the two machines must run the same operating system is a requirement when using high availability.

If we assume the above, then the three scripts need to be configured as follows:

1. goActive script

```
#!/bin/ksh
#
#goActive script
#
#This script is executed when a Network Dispatcher
#goes into the 'active' state and begins
#routing packets.
# CLUSTER=9.67.123.8
SECCLUSTER=10.0.0.8
INTERFACE=en0
SECINTERFACE=en1
# echo "Deleting loopback aliases"
ifconfig lo0 delete $CLUSTER
ifconfig lo0 delete $SECCLUSTER
# echo "Adding device aliases"
ifconfig $INTERFACE alias $CLUSTER netmask 255.255.248.0
ifconfig $SECINTERFACE alias $SECCLUSTER netmask 255.255.255.0
```

Figure 285. goActive Script

2. goInOp script

```
#!/bin/ksh
#
#goInOp script
#
#This script is executed when a Network Dispatcher
#executor is stopped (and before the executor is initially started).
# CLUSTER=9.67.123.8
SECCLUSTER=10.0.0.8
INTERFACE=en0
SECINTERFACE=en1
# echo "Removing all loopback aliases"
ifconfig lo0 delete $CLUSTER
ifconfig lo0 delete $SECCLUSTER
# echo "Removing all device aliases"
ifconfig $INTERFACE delete $CLUSTER
ifconfig $SECINTERFACE delete $SECCLUSTER
```

Figure 286. goInOp script

3. goStandby script

```

#!/bin/ksh
#
#goStandby script
#
#This script is executed when a Network Dispatcher goes into the 'standby'
state,
#monitoring the health of the active machine,
#but routing no packets.
# CLUSTER=9.67.123.8
SECCLUSTER=10.0.08
INTERFACE=en0
SECINTERFACE=en1
# echo "Deleting the device aliases"
ifconfig $INTERFACE delete $CLUSTER
ifconfig $SECINTERFACE delete $SECCLUSTER
# echo "Adding loopback aliases"
ifconfig lo0 alias $CLUSTER netmask 255.255.248.0
ifconfig lo0 alias $SECCLUSTER netmask 255.255.255.0

```

Figure 287. goStandby script

4.13.2 Setting the Rules for IBM eNetwork Firewall

The next step in configuration is setting the rules for eNetwork Firewall to allow the Dispatcher to communicate with itself and the other machine.

The Dispatcher is a client/server application. The initial program that is started is `ndserver`, all the other commands are then issued using `ndcontrol`. When an `ndcontrol` command is issued, the `ndcontrol` program attempts to open a socket connection with the `ndserver` program. If it succeeds, the command is issued and the results returned. The source port for these TCP/IP packets will be a random port greater than 1023 and the destination port, by default, is 10003. This port can be changed by the user if deemed necessary, and instructions for that are located in 4.3.4, "TCP Ports Used by the Dispatcher" on page 217. For example, if using the default, the socket might be opened using a TCP/IP connection from port 1048 to port 10003. The Dispatcher opens this socket on the default interface. The default interface is the one on which the host name of the machine is defined. The entire socket connection is taking place along the default interface so the rule can be explicit as to only allow communication along port 10003 from the interface back to itself.

One Note of Caution

The default interface may be the non-secure interface, and some firewall administrators may take offense to this. The risk is minimal, however, due to the fact that the socket need only be allowed to port 10003 and only along the default interface.

The Dispatcher synchronizes the primary and backup machines using TCP/IP packets, called heartbeats. A rule has to be made to allow these heartbeats to be transmitted from one machine to the other. Heartbeat configuration is vitally

important in order to ensure a high availability solution. If the two Dispatcher machines lose contact with each other they will both assume that the other machine has failed and will assume routing duties for the cluster addresses. If this state occurs there will be network errors due to the fact that there are multiple machines on the network answering to the same addresses. In order to prevent this, it is recommended to have multiple heartbeats, by having one heartbeat for each interface in the machine. If there are only two interfaces in the machine, one secure and one non-secure, then one of the heartbeats needs to be sent along the non-secure interface. Firewall administrators may not like opening up another port on the non-secure interface, although the risk is small.

The port used by the heartbeat is defined by the user, and the TCP/IP packet will be constructed using this port as its source and destination. For example, if the user defines the heartbeat port as 12345, then the TCP/IP packet will be issued from port 12345 and to port 12345. This is convenient as it allows the rule on the firewall to be very precise. If one does not want to open this connection along the non-secure interface, the solution is to have multiple network cards on the secure network side and exchange heartbeats along each of those.

The last rule that needs to be opened on the firewall is for reachability. The Dispatcher has a feature called reachability that allows the user to define a series of IP addresses that the Dispatcher machines will attempt to ping; these machines are called reach targets. If the ping is successful, the Dispatcher assumes that it can reach those machines. For example, if the backup machine can reach more of the machines than the primary, it is likely that one of the network interfaces on the primary is no longer working and a fail-over will occur.

The firewall needs to be configured to allow each of these pings to reach their destination. The reachability ping is simply a standard ICMP ping from port 8 to port 0. Since the IP addresses are chosen by the user, the firewall configuration can be precise on allowing pings only to those specified machines if necessary. It is not necessary to allow inbound pings to the firewall.

Once the ports have been properly configured on the firewall, the Dispatcher can be configured. One last decision has to be made, and that is how recovery should be implemented. The Dispatcher has two modes, automatic and manual:

1. In automatic mode, if the primary fails, the backup takes over. When the primary recovers it will automatically take over from the backup machine.
2. In manual mode, if the primary fails, the backup takes over. When the primary recovers it will not takeover unless the backup fails, or given an explicit command to do so.

The decision on which mode to use will depend on the individual customer situation.

The reasons for fail-over are as follows. If a machine is no longer receiving heartbeat information, it assumes the other machine has failed, and a failover will occur. If the backup machine is getting responses from more reach targets than the primary, a failover will occur. In all other situations the two machines should stay synchronized and be transparent to the firewall.

4.13.3 Scenario Implementation

We show now a sample configuration setup and give a concrete example of the steps mentioned in this chapter.

The following table summarizes the high availability settings:

Table 11. High Availability Settings

Firewall Machine	FW1	FW2
IP Address for Non-Secure Network	9.67.123.4	9.67.123.5
IP Address for Secure Network	10.0.0.4	10.0.0.5
Heartbeat Port	12345	12345
Reach Target Non-Secure	9.67.123.6	9.67.123.6
Reach Target Secure	10.0.0.6	10.0.0.6

The following list shows the configuration steps:

1. Configure FW1 and FW2 to allow communication from a port greater than 1023 to port 10003 along the default interface.
2. Allow communication between FW1 and FW2 using port 12345 as source and destination on the secure and non-secure interfaces.
3. Allow ICMP pings outbound from the non-secure interface to 9.67.123.6 and from the secure interface to 10.0.0.6.
4. Edit goActive, goInOp and goStandby scripts to alias the cluster addresses for the secure and non-secure interfaces.
5. Start the ndserver program on FW1 and configure the Dispatcher to provide high availability.

This is the sequence of commands to issue on FW1:

```
ndserver start
ndcontrol executor start
ndcontrol highavailability heartbeat add 9.67.123.4 9.67.123.5
ndcontrol highavailability heartbeat add 10.0.0.4 10.0.0.5
ndcontrol highavailability backup add primary automatic 12345
ndcontrol highavailability reach add 9.67.123.6
ndcontrol highavailability reach add 10.0.0.6
ndcontrol manager start
ndcontrol file save HAFirewall
```

6. Start the ndserver program on FW2 and configure the Dispatcher to provide high availability.

This is the sequence of commands to issue on FW2:

```
ndserver start
ndcontrol executor start
ndcontrol highavailability heartbeat add 9.67.123.5 9.67.123.4
ndcontrol highavailability heartbeat add 10.0.0.5 10.0.0.4
ndcontrol highavailability backup add backup automatic 12345
ndcontrol highavailability reach add 9.67.123.6
ndcontrol highavailability reach add 10.0.0.6
ndcontrol manager start
ndcontrol file save HAFirewall
```

7. Direct all non-secure clients to 9.67.123.8 and all secure clients to 10.0.0.8.

Note the following:

- The following command issued in steps 5 on page 344 and 6 on page 344, would fail if step 1 on page 344 is not executed:

```
ndcontrol executor start
```

- The following command also issued in steps 5 on page 344 and 6 on page 344, will allow the configuration to be reloaded quickly:

```
ndcontrol file save HAFirewall
```

As you can see, we have shown how to issue the high availability configuration through the command line. Refer to 4.12, “Dispatcher High Availability Scenario” on page 314 to see how to perform the same steps by using the eNetwork Dispatcher GUI.

If either machine is stopped or rebooted, then the Dispatcher has to be reconfigured. It does not start automatically. In order to reconfigure it perform the following two steps, assuming you saved the configuration file as HAFirewall, as indicated in steps 5 on page 344 and 6 on page 344:

```
ndserver start  
ndcontrol file load HAFirewall
```

Part 2. IBM WebSphere Performance Pack Scenarios

Chapter 5. Enterprise Campus Scenario

This chapter gets very specific about how to implement an enterprise campus architecture. A graphical representation of the enterprise campus scenario we built is shown in the following figure:

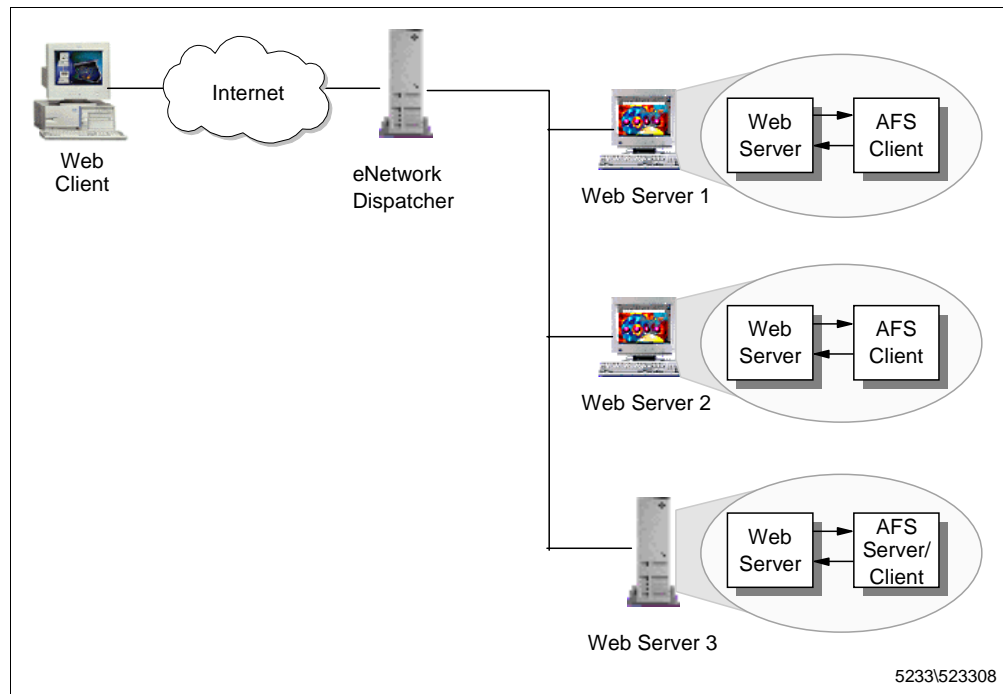


Figure 288. Enterprise Campus Scenario

As you can see, this scenario uses two components of IBM WebSphere Performance Pack:

1. Load Balancing component, also known as IBM eNetwork Dispatcher
2. File Sharing component, also known as AFS File System

The content is actually stored only in the first AFS machine, which is simultaneously an AFS server and client. The two AFS client machines that are part of the environment as well simply use the data shared by the first AFS machine.

The Web site implemented with this solution is very powerful, since it is built up with three clustered Web servers that share the same data and a load balancing machine that distributes the workload between the three Web servers.

This solution keeps data between all the Web servers consistent, while making it easy to update the data. It offers the opportunity of having the exact directory structure so that Web servers can have the same configuration files. Moreover this is a scalable solution in that you have the ability to add more Web servers as needed, without stopping the service.

The architecture of an enterprise campus Web site also benefits from *Web server high availability*, guaranteed by the load balancing machine. If one of the Web

servers belonging to the cluster goes out of service, the service is not interrupted, since the other two Web servers keep working. You can also easily implement *ISS high availability* if you decide to use the ISS function in your architecture (see 4.8.1, “ISS Configuration File” on page 295).

In our scenario, the client machine and the Web site belonged to two different token-ring LANs at 16 Mbps, separated by a router machine, which was a RISC/6000 provided with two token-ring network interfaces. All the details on how we built our environment are described in Appendix A, “Network Configuration of the Scenario Environments” on page 405.

We built the enterprise campus scenario in two steps:

1. First we implemented a *traditional scenario*, where the Web site was composed of a single Web server machine, without any use of IBM WebSphere Performance Pack.
2. Then we enhanced the traditional scenario and built a complete *enterprise campus scenario* using the Load Balancing and File Sharing components of IBM WebSphere Performance Pack, as shown in Figure 288 on page 349.

In both the steps we used WebStone to generate the workload (see Appendix B, “Scenario Basic Configuration” on page 411). We also extracted the output information generated by WebStone to measure the Web site performances in both scenarios to see how performances increase when using IBM WebSphere Performance Pack. Notice that the results of our measurements can be considered valid only keeping in mind the limitations of our environment. As we discuss in the box “Value of Performance Measurements” on page 417, when we built our scenarios, our main purpose was *not* to measure the performances of the Web sites we created. Our scenarios were aimed at demonstrating how the various components of IBM WebSphere Performance Pack can be combined together to provide complex and powerful architectures. We implemented such architectures and we tried them simulating a workload with WebStone.

Performance measurements in an environment as limited as ours do not have a real value. We could have used more powerful machines and a larger number of clients, tuned our machines to get a better utilization of CPU and a faster disk access, used a faster network, etc. However, keeping in mind these limitations, we think it is interesting to see how the performance of a certain Web site, although not optimized, increase when using IBM WebSphere Performance Pack.

5.1 First Step: Traditional Scenario

As we mentioned in 1 on page 350, this scenario does not make use of any of the components of IBM WebSphere Performance Pack. We implemented this scenario as a first step before building the complete enterprise campus scenario. This scenario has a value also in terms of performance, because we will compare the performance of this scenario with the performances measured while using IBM WebSphere Performance Pack.

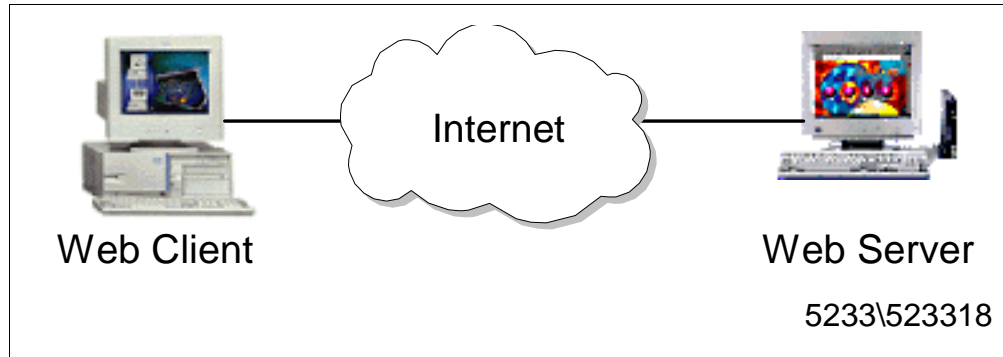


Figure 289. Graphical Representaion of the Traditional Scenario

The architecture of this basic scenario was very simple; a Web client submits its requests to a Web server and the Web server serves the responses back to the client. The two machines are located in two different networks, which in our environment were separated by an IP router.

The following table describes the hardware, software and network architecture of the scenario environment:

Table 12. Traditional Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	aixncf157	172.16.1.5	AIX 4.3.1	Web Client
IBM RS/6000 43P	rs600012	I) 172.16.1.1 E) 9.24.104.124	AIX 4.3.1	IP Router
IBM PC 365	computer1	9.24.104.223	Windows NT Server 4.0	Web Server

The role of the Web client was provided by WebStone 1.1, while the Web server functionality was provided by Lotus Domino Go Webserver 4.6.2.5. All the details on the configuration of the environment shown in the above table can be found in Appendix A, "Network Configuration of the Scenario Environments" on page 405. The configuration of WebStone can be found in Appendix B, "Scenario Basic Configuration" on page 411.

As we explain in B.3.3, "WebStone Configuration Files" on page 413, the workload was light dynamic, meaning it was made up of a mix of static and dynamic pages.

This is the testbed file we used in this scenario:

```

#Testbed file used for testing only one Web Server
ITERATIONS="1"
MINCLIENTS="20"
MAXCLIENTS="100"
CLIENTINCR="10"
TIMEPERRUN="2"
SERVER="computer1"
PORTNO=80
WEBSTONEROOT="/home/guest/WebStone-1.1"
CLIENTS="aixncf157"
CLIENTACCOUNT=root
CLIENTPASSWORD=pistoia
TMPDIR=/tmp
RCP="rcp"
RSH="rsh"

```

Figure 290. WebStone testbed File

The WebStone filelist file for light dynamic workload is shown in the following figure:

```

#Modified Silicon Surf model; added light dynamic content use
8
40 2
/afs/webstone/html/file2k.html
/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072
25 2
/afs/webstone/html/file1k.html
/afs/webstone/html/file5k.html
15 2
/afs/webstone/html/file4k.html
/afs/webstone/html/file6k.html
5 1
/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=7168
4 4
/afs/webstone/html/file8k.html
/afs/webstone/html/file9k.html
/afs/webstone/html/file10k.html
/afs/webstone/html/file11k.html
4 5
/afs/webstone/html/file12k.html
/afs/webstone/html/file14k.html
/afs/webstone/html/file15k.html
/afs/webstone/html/file17k.html
/afs/webstone/html/file18k.html
6 1
/afs/webstone/html/file33k.html
1 1
/afs/webstone/html/file200k.html

```

Figure 291. WebStone filelist File

After running WebStone, we got the following performance output:

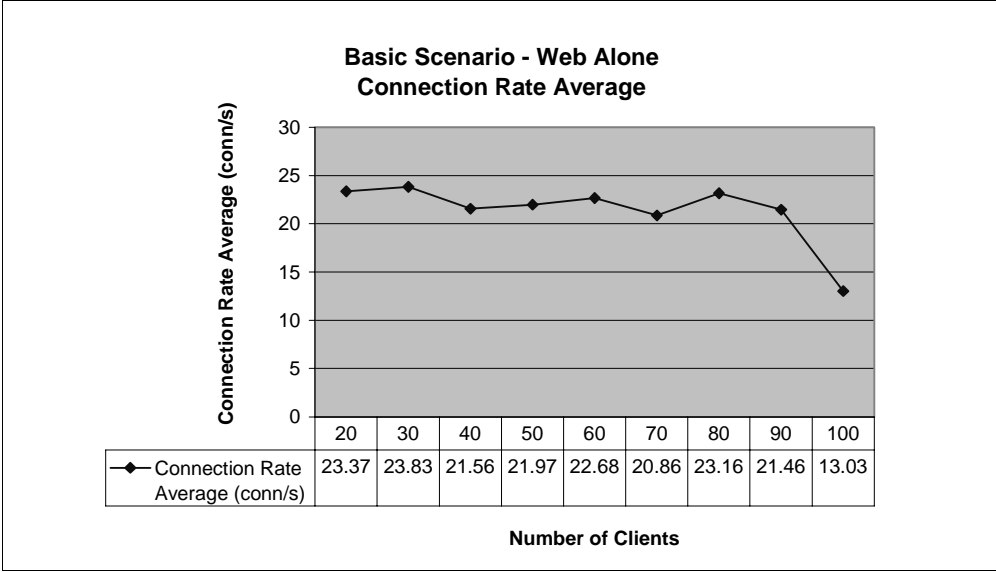


Figure 292. Basic Scenario - Web Alone - Connection Rate Average

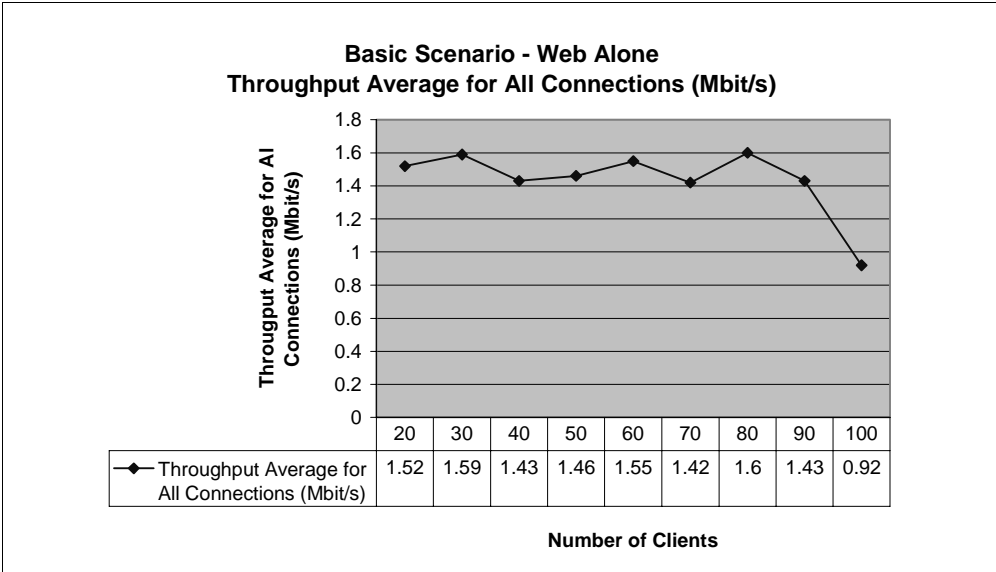


Figure 293. Basic Scenario - Web Alone - Throughput Average for All Connections

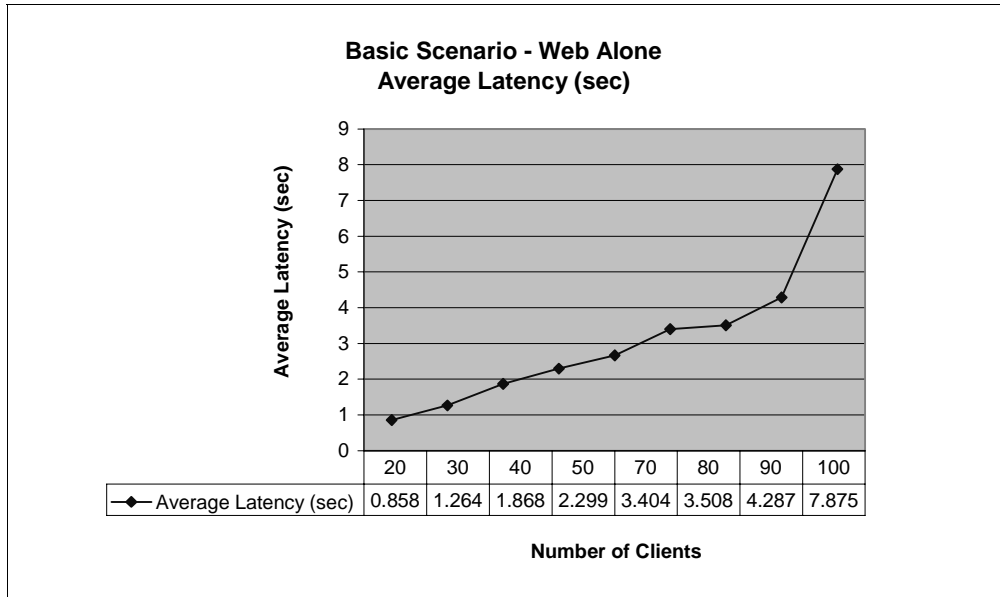


Figure 294. Basic Scenario - Web Alone - Average Latency

The above figures demonstrate that performances decrease dramatically when the number of clients increases. For example, you can see that the average latency, which is the average a client waits for data to be returned, is close to 8 seconds when 100 concurrent clients are submitting requests to the Web server.

We also verified that even when only 20 clients were accessing the Web site simultaneously, the CPU utilization of the Web server machine was 100%, as shown in the following figure:

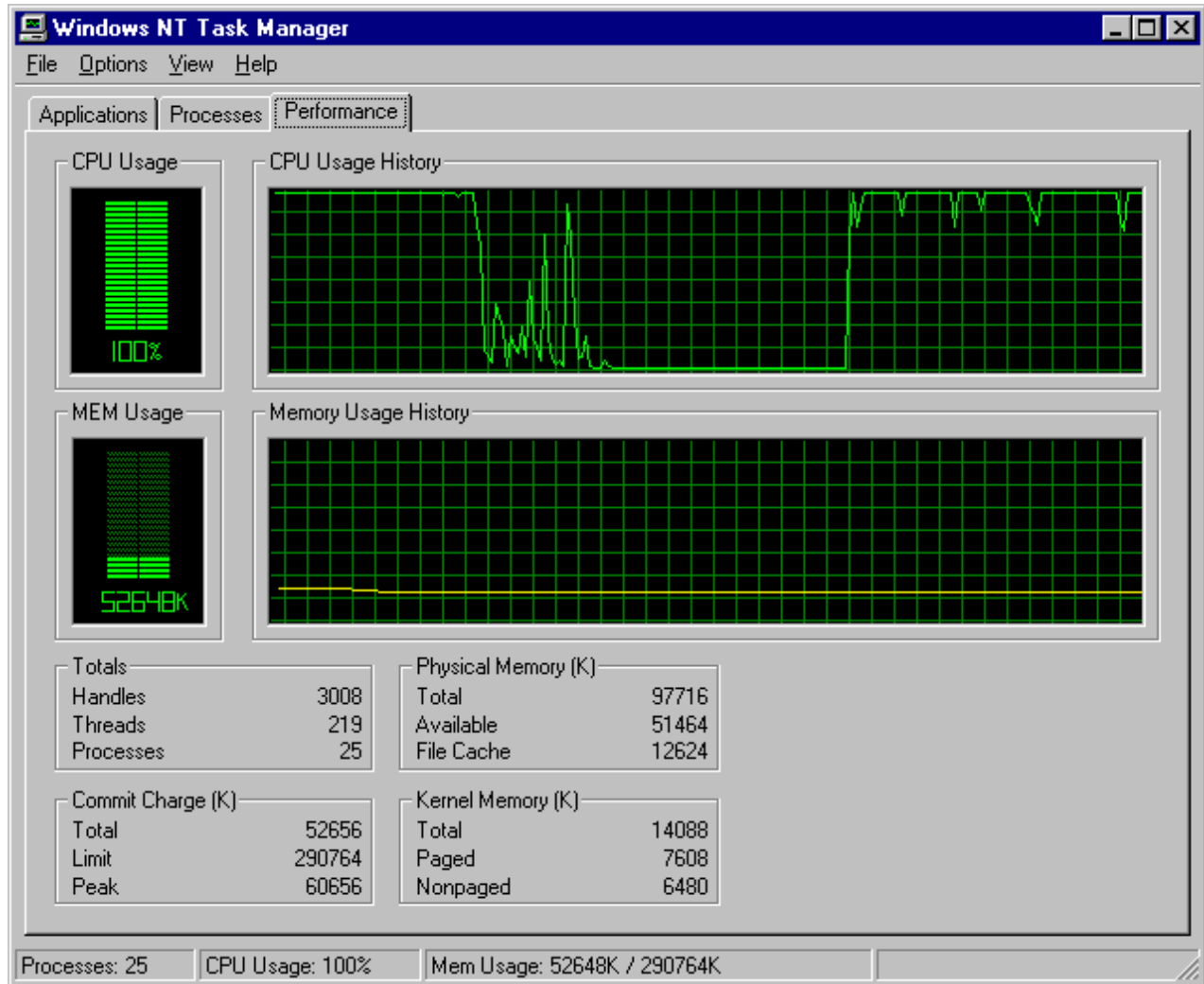


Figure 295. Windows NT Task Manager on the Web Server Machine

Notice that the Web server machine had a CPU of 166 MHz.

5.2 Second Step: Enterprise Campus Complete Scenario

In this section we show you how to enhance the traditional scenario shown in 5.1, “First Step: Traditional Scenario” on page 350. A graphical representation of the flow of this scenario is shown in the following diagram:

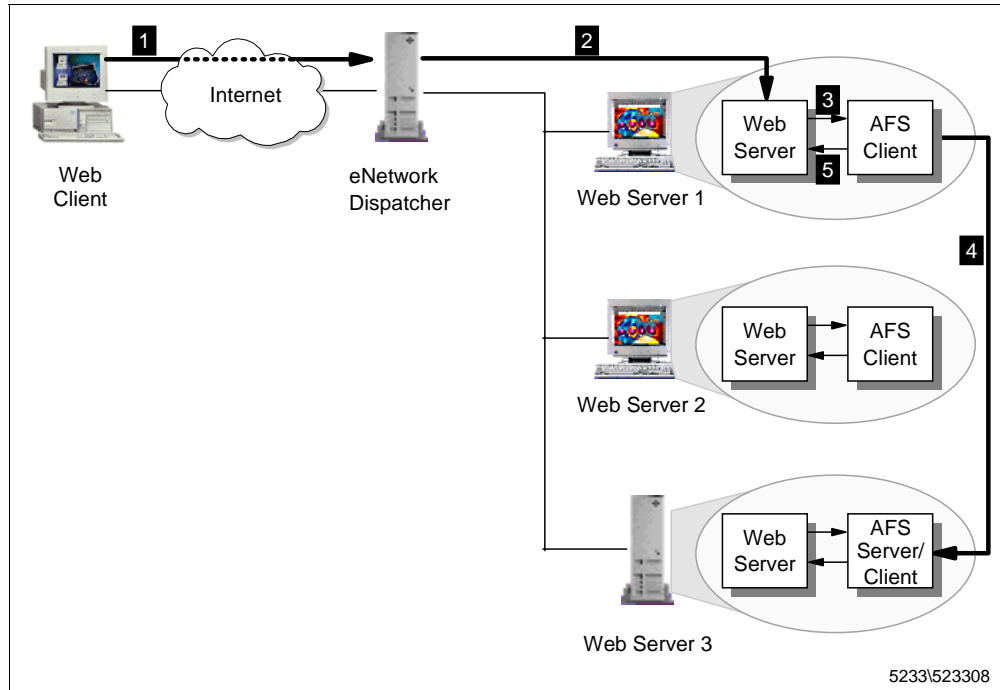


Figure 296. Flow Diagram of the Enterprise Campus Scenario

The flow of the enterprise campus scenario that we have represented in the above diagram is described in the following list:

1. The Web client software issues a URL request using the cluster address of the Web site (for instance, <http://9.24.104.105/afs/webstone/html/file2k.html>). This request is intercepted by the Dispatcher machine.
2. The Dispatcher machine selects the appropriate Web server (in this example, Web Server 1) and forwards the request to it.
3. The httpd daemon running on Web Server 1 interpretes the HTTP request and issues a local I/O call that is intercepted by the AFS Cache Manager to retrieve the data.
4. The Cache Manager retrieves the data from its local cache if the requested data is present and up-to-date, or requests it from the AFS server.
5. Upon retrieving data, the Cache Manager gives it back to the httpd daemon that is able to honor the client's request directly, without passing through the Dispatcher machine.

The architecture shown in the above diagram is an enhancement from the basic architecture described in 5.1, "First Step: Traditional Scenario" on page 350, in that the single Web server on the server-side has been now replaced by a complex and reliable architecture:

- In an enterprise campus scenario, the AFS server maintains the file name space, provides the AFS clients with Web contents, when these are requested, replicates contents in a non-disruptive way, and maintains synchronization with mirrored contents and cached data in the AFS client machine. The use of the File Sharing component of IBM WebSphere

Performance Pack guarantees that the three Web servers can share the same contents in a non-disruptive way.

- The Load Balancing component of IBM WebSphere Performance Pack increases the Web site performances, in that it is able to forward the client's requests to the next best Web server to handle the load. Since the Web server responds directly to the client, without any need to involve the load balancing machine again, a faster network can be used for the response physical path.
- Combining the file sharing and load balancing functions in the same environment guarantees Web server high availability, meaning that if one of the Web servers should fail the service would not be interrupted and the end user would not experience any problems.

The hardware, software and network architecture of the enterprise campus scenario we implemented is shown in the following table:

Table 13. Complete Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	aixncf157	172.16.1.5	AIX 4.3.1	Web Client
IBM RS/6000 43P	rs600012	I) 172.16.1.1 O) 9.24.104.124	AIX 4.3.1	IP Router
IBM RS/6000 43P	1) rs600023 2) clusterend	1) 9.24.104.128 2) 9.24.104.105	AIX 4.3.1	Load Balancing
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3.1	AFS Server and Client, Web Server
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	AFS Client, Web Server
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	AFS Client, Web Server

Notice that the separation between the client-side and server-side networks was achieved in our environment by using an IP router.

The role of the Web client was provided by WebStone 1.1, while the Web server functionality was provided by Lotus Domino Go Webserver 4.6.2.5. All the details on how we configured the network environment are described in Appendix A., "Network Configuration of the Scenario Environments" on page 405. The configuration of the Web client machine to run WebStone is shown in Appendix B., "Scenario Basic Configuration" on page 411.

We defined a cluster of three servers on the TCP port 80, since we wanted the Dispatcher to load-balance only HTTP requests.

In order to repeat this same experience you must perform the following steps:

1. Set up all the machines (see Appendix A., "Network Configuration of the Scenario Environments" on page 405 and Appendix B., "Scenario Basic Configuration" on page 411).
2. Install and configure the load balancing component (see 4.5, "Installation of the Load Balancing Component" on page 229 and 4.6.3, "Dispatcher Configuration" on page 248).

We did not use the ISS function in this architecture, but if you want to implement ISS in your environment you can refer to 4.8, “Load Balancing Scenario Using the Dispatcher and ISS Functions” on page 294.

3. Configure the Web servers to be part of the clustered Web site (see 4.6.4, “TCP Servers Configuration” on page 267).
4. Install and configure the File Sharing component on the Web server machines (see 2.4, “Installation and Configuration of the File Sharing Component” on page 21).

The Dispatcher configuration we implemented is shown in the following four figures:

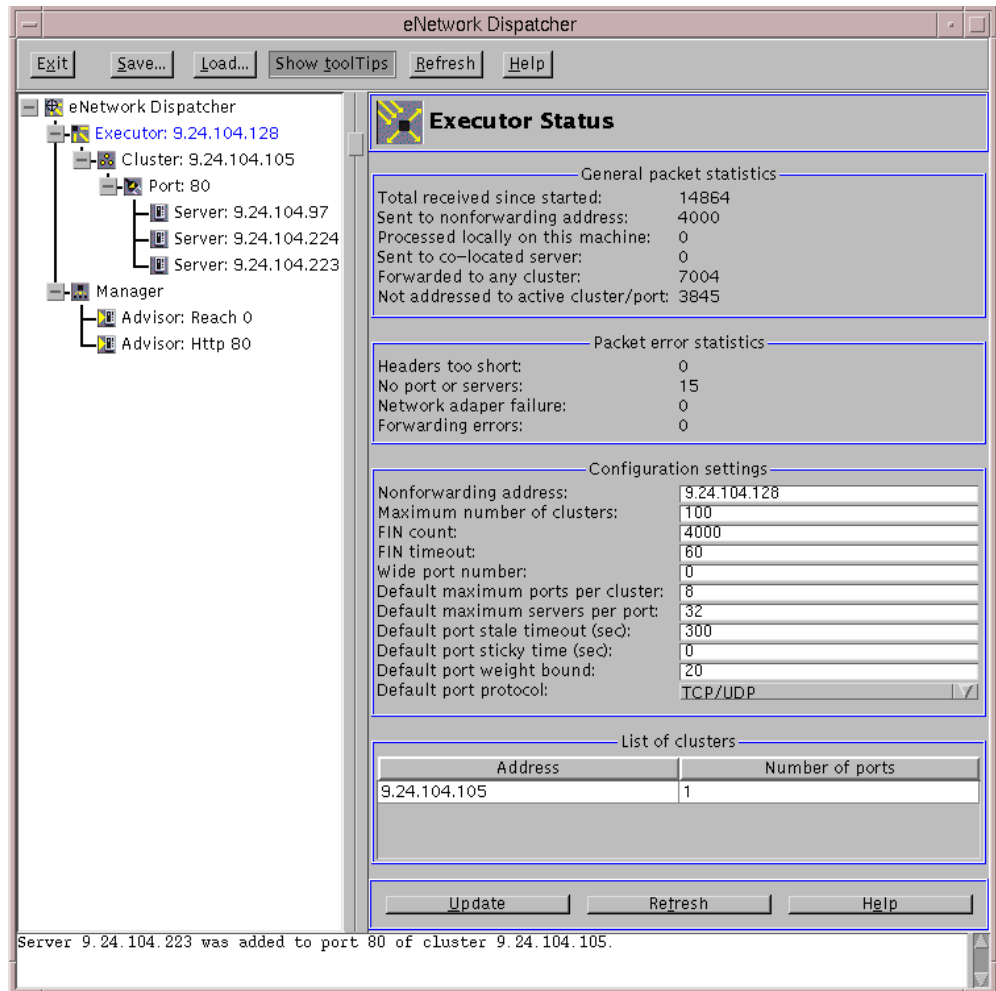


Figure 297. Dispatcher Configuration - Executor Status

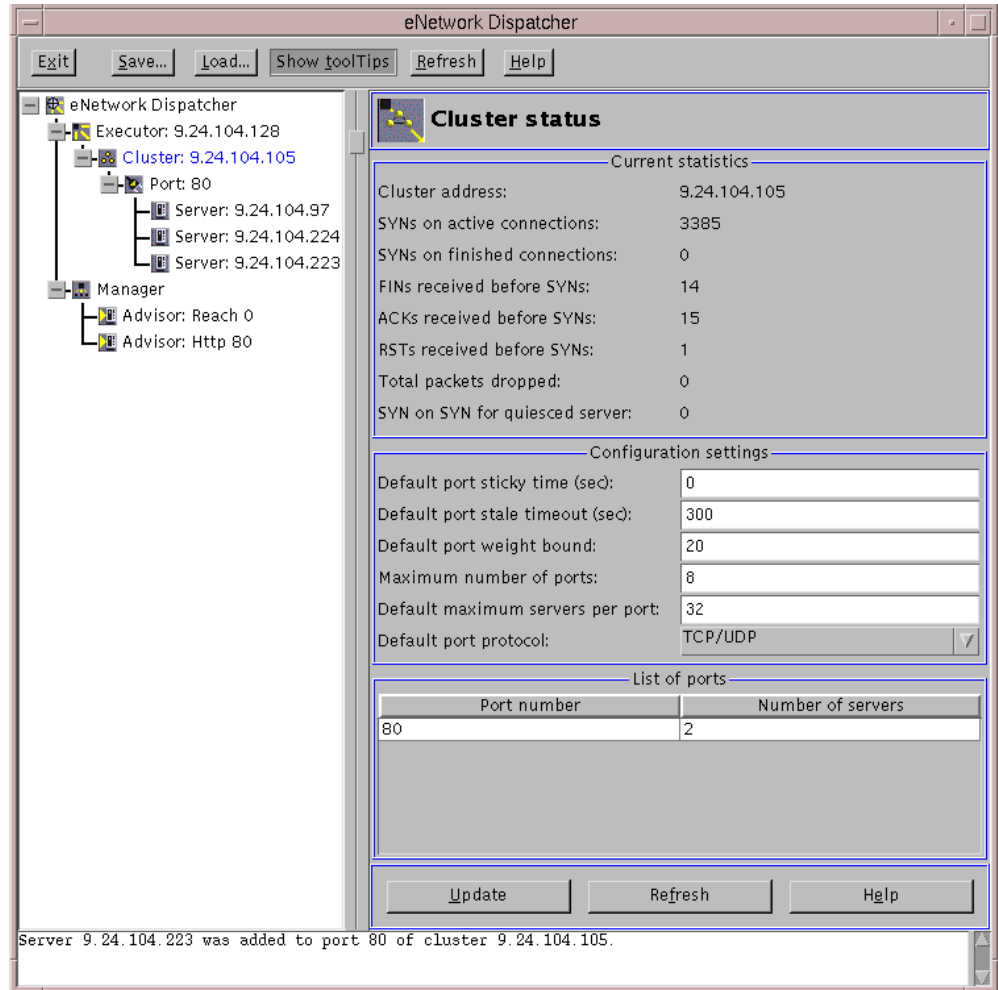


Figure 298. Dispatcher Configuration - Cluster Status

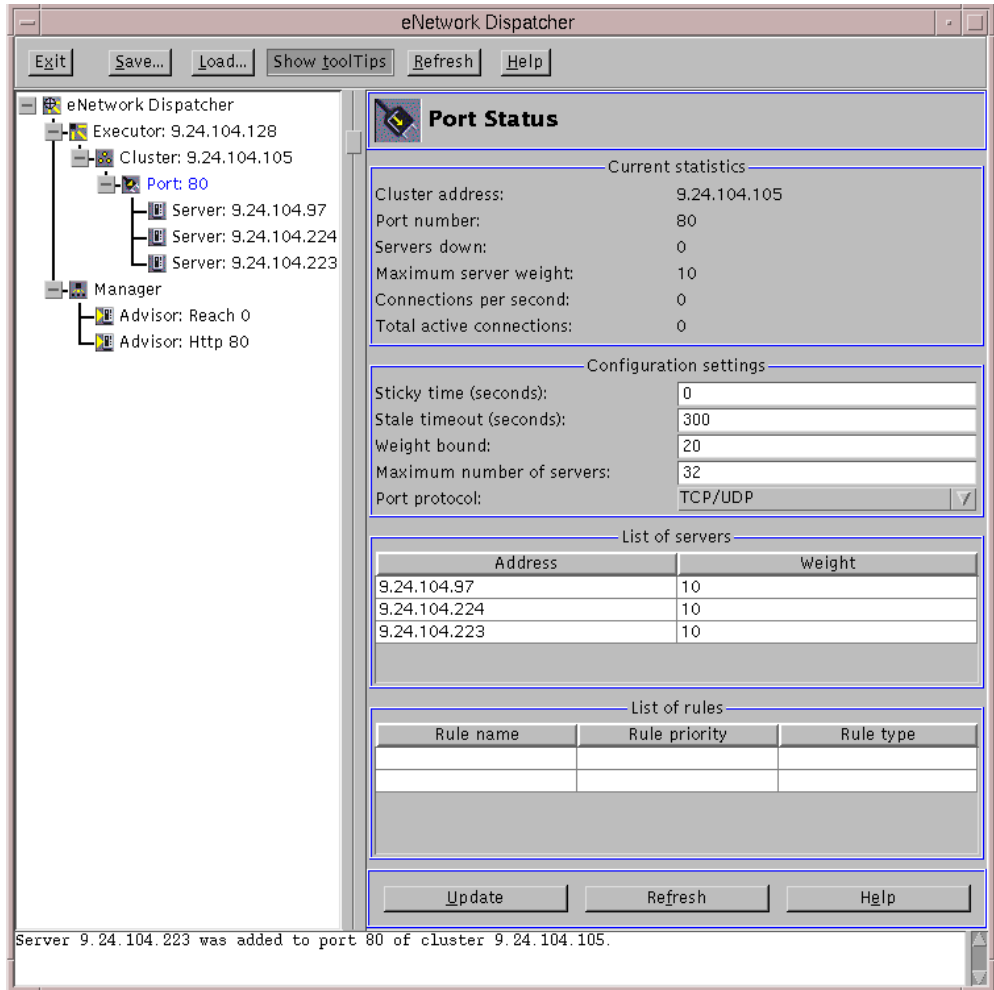


Figure 299. Dispatcher Configuration - Port Status

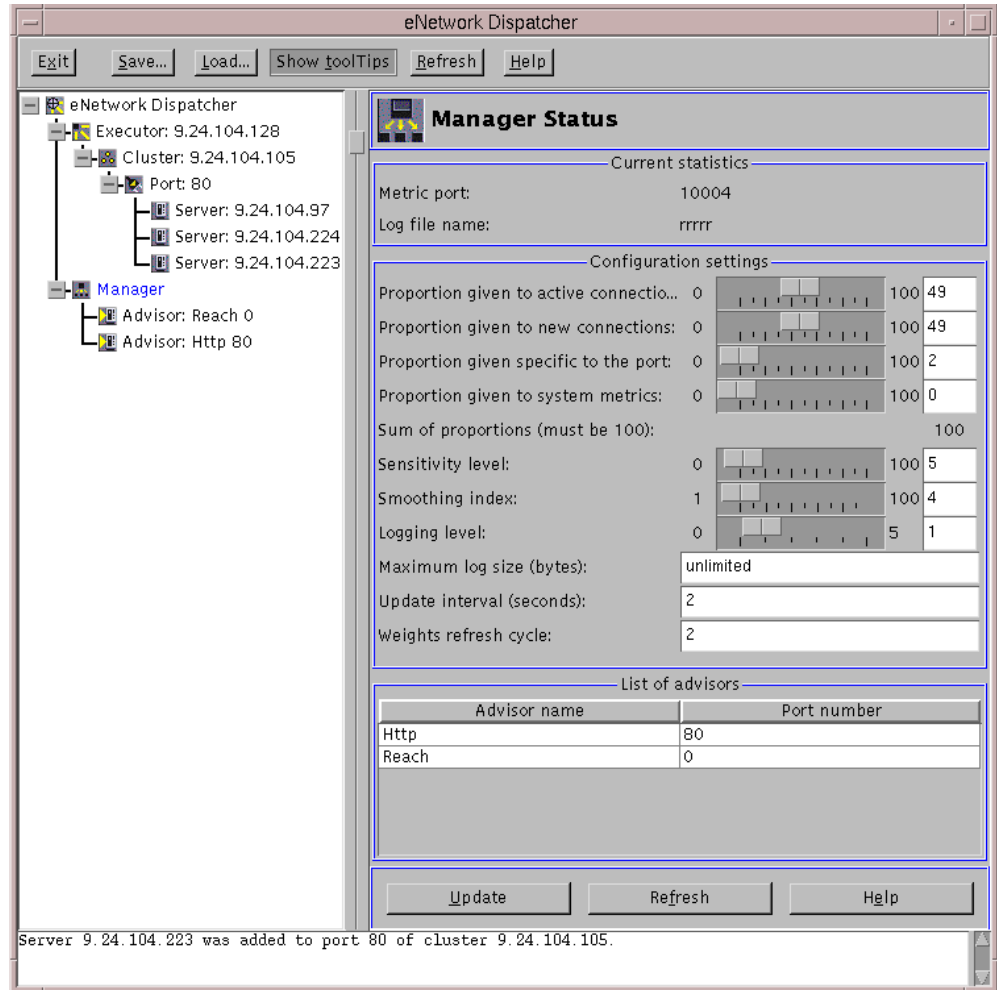


Figure 300. Dispatcher Configuration - Manager Status

As we explained in Appendix B.3.3, “WebStone Configuration Files” on page 413, the workload was light dynamic, meaning it was made up of a mix of static and dynamic pages.

This is the testbed file we used in this scenario:

```

#Testbed file used for testing Dispatcher plus three Web Servers
ITERATIONS="1"
MINCLIENTS="20"
MAXCLIENTS="100"
CLIENTINCR="10"
TIMEPERRUN="2"
SERVER="clusterend"
PORTNO=80
WEBSTONEROOT="/home/guest/WebStone-1.1"
CLIENTS="aixncf157"
CLIENTACCOUNT=root
CLIENTPASSWORD=pistoia
TMPDIR=/tmp
RCP="rcp"
RSH="rsh"

```

Figure 301. WebStone testbed File

Notice that the `SERVER` parameter is set to `clusterend`, which was the host name associated to the cluster address in our environment.

The WebStone filelist file for light dynamic workload is shown in the following figure:

```

#Modified Silicon Surf model; added light dynamic content use
8
40 2
/afs/webstone/html/file2k.html
/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072
25 2
/afs/webstone/html/file1k.html
/afs/webstone/html/file5k.html
15 2
/afs/webstone/html/file4k.html
/afs/webstone/html/file6k.html
5 1
/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=7168
4 4
/afs/webstone/html/file8k.html
/afs/webstone/html/file9k.html
/afs/webstone/html/file10k.html
/afs/webstone/html/file11k.html
4 5
/afs/webstone/html/file12k.html
/afs/webstone/html/file14k.html
/afs/webstone/html/file15k.html
/afs/webstone/html/file17k.html
/afs/webstone/html/file18k.html
6 1
/afs/webstone/html/file33k.html
1 1
/afs/webstone/html/file200k.html

```

Figure 302. WebStone filelist File

We want to show you now the intense activity of the three Web servers as soon as we started WebStone and the Dispatcher started its load balancing function. The following figures show some of the entries registered in the httpd-log files of the three Web servers, in a few seconds' interval. These figures were captured on the Web server machines rs600030, wtr05195 and computer1 respectively:

```

172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file1k.html HTTP/1.0" 200 1024
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file6k.html HTTP/1.0" 200 6144
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file11k.html HTTP/1.0" 200 11264
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file14k.html HTTP/1.0" 200 14336
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file8k.html HTTP/1.0" 200 8192
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file5k.html HTTP/1.0" 200 5120
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file1k.html HTTP/1.0" 200 1024
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file33k.html HTTP/1.0" 200 33792
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file4k.html HTTP/1.0" 200 4096
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=7168 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file5k.html HTTP/1.0" 200 5120
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file5k.html HTTP/1.0" 200 5120
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file18k.html HTTP/1.0" 200 18432
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:32 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file11k.html HTTP/1.0" 200 11264
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file1k.html HTTP/1.0" 200 1024
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file5k.html HTTP/1.0" 200 5120
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072 HTTP/1.0" 200
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file2k.html HTTP/1.0" 200 2048
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file4k.html HTTP/1.0" 200 4096
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file4k.html HTTP/1.0" 200 4096
172.16.1.5 - - [28/Sep/1998:13:52:33 +0500] "GET /afs/webstone/html/file6k.html HTTP/1.0" 200 6144

```

Figure 303. httpd-log File on the Web Server rs600030

In the following three figures we show you the performance diagrams of this scenario. In order to show you how performances increase when using IBM WebSphere Performance Pack, we also report on the same figures the diagrams related to the basic Web site, described in 5.1, “First Step: Traditional Scenario” on page 350:

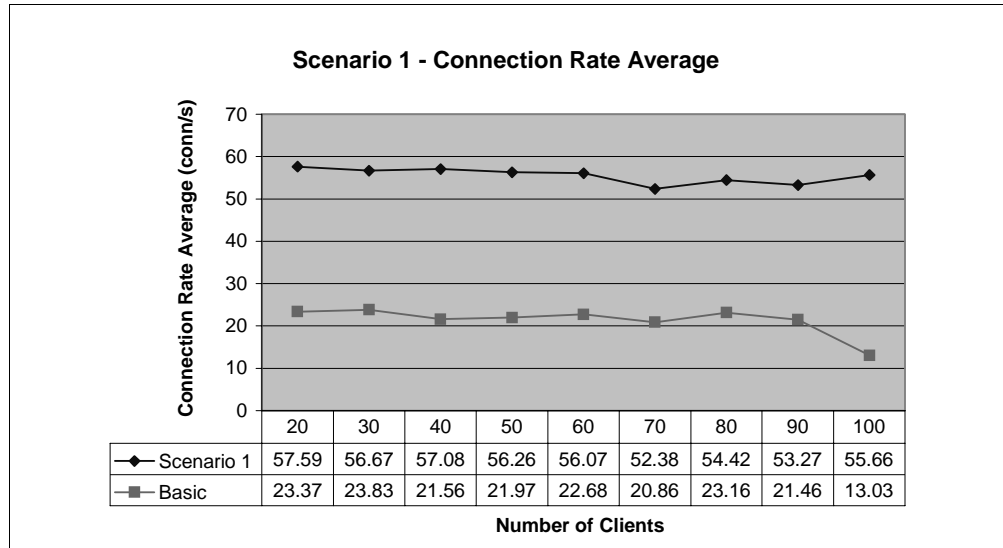


Figure 306. Scenario 1 - Connection Rate Average

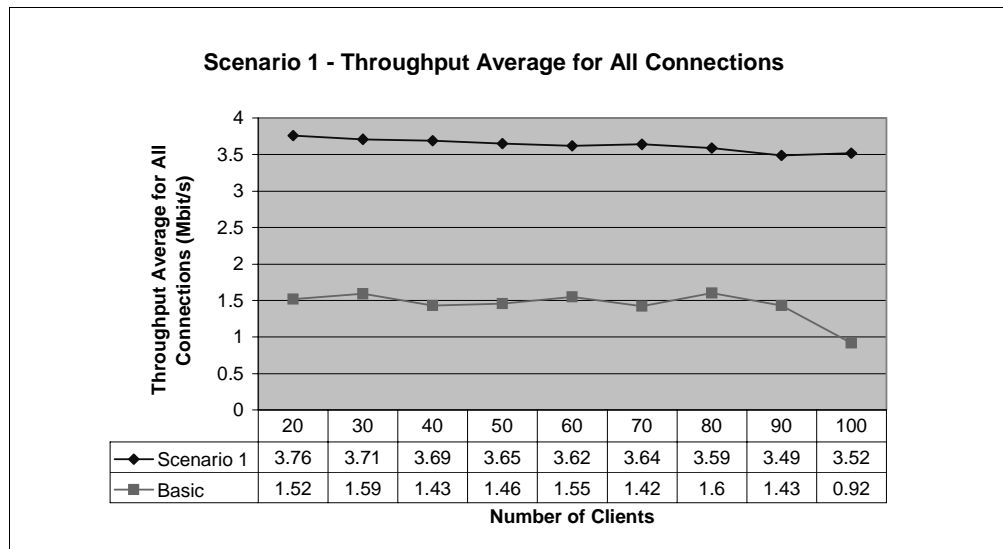


Figure 307. Scenario 1 - Throughput Average for All Connection

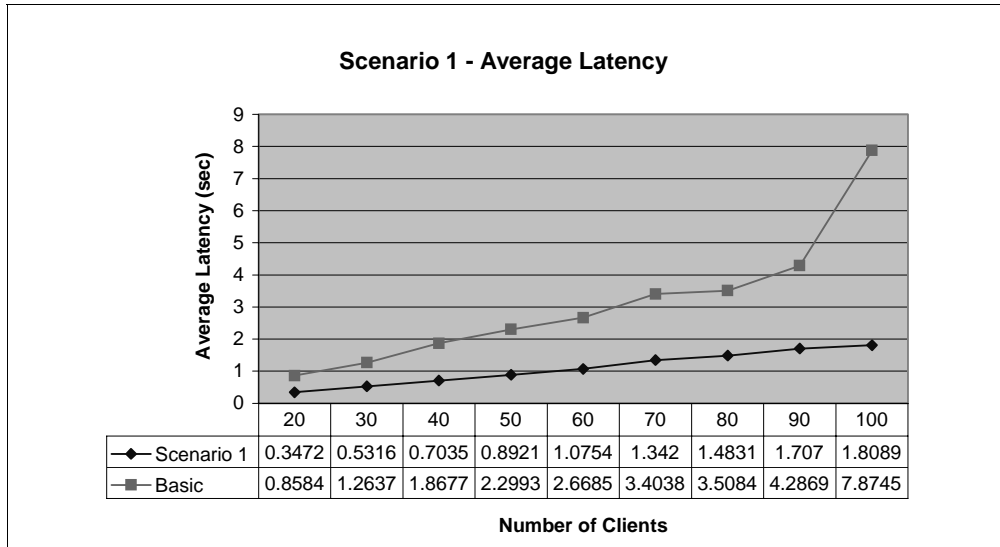


Figure 308. Scenario 1 - Average Latency

As you can see, if a Web site implements an enterprise campus architecture similar to the one we have just shown, when 100 concurrent clients access the same Web site generating a light dynamic workload (25% of dynamic pages that invoke a CGI-Bin program, 75% of static HTML pages), the average client wait for data to be returned is less than 2 seconds, 6 seconds less than implementing a traditional Web site.

We also verified that the average CPU utilization was 40% on each Web server, while it reached 100% in a traditional Web site (see Figure 295 on page 355).

Chapter 6. Regional and Local Access ISP Scenario

This chapter shows how to install, configure and customize all the IBM WebSphere Performance Pack components in order to implement a regional and local access Internet Service Provider (ISP) architecture. A graphical representation of the scenario we built is shown in the following figure:

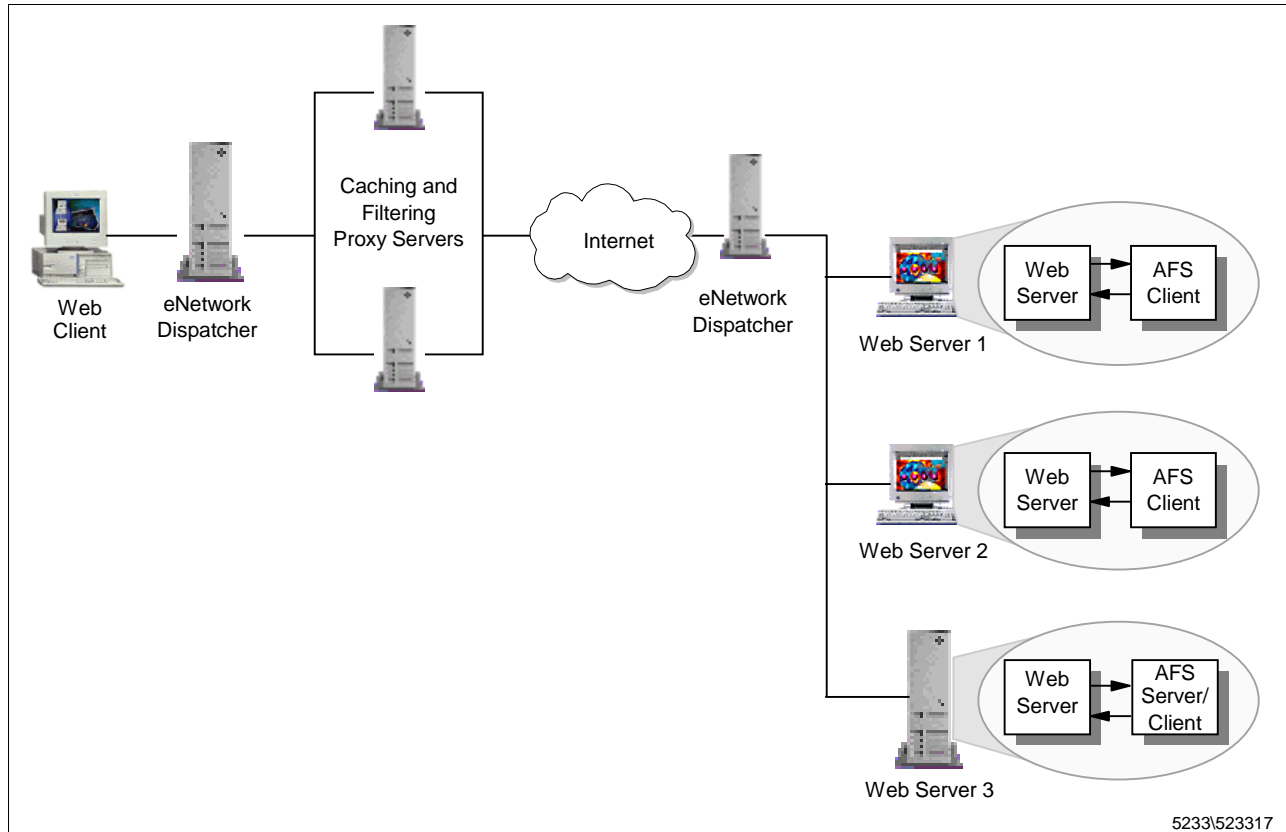


Figure 309. Regional and Local Access ISP Scenario Architecture

As you can see, this architecture makes use of all the components of IBM WebSphere Performance Pack:

1. Requests coming from client machine are intercepted by a caching and filtering proxy server, which serves the response directly to the client if the requested Web page has already been requested and cached, or it forwards the client's request to the Web site, receives the Web page, caches it if is cacheable and then serves it back to the client. The caching and filtering proxy server can also apply content filtering at the proxy server level, according to the PICS specifications.
2. The Load Balancing component is used not only to distribute the load between different Web servers, but also to balance the load between two or more caching and filtering proxy servers.
3. The File Sharing component allows the Web server machines to share the same Web content in a non-disruptive way. The AFS file system keeps data between all the Web servers consistent, while making it easy to update the

data. It offers the opportunity of having the exact directory structure so that Web servers can have the same configuration files.

Balancing two or more caching and filtering proxy servers with a load balancing machine guarantees *caching and filtering high availability*, in that if one of the clustered proxy servers should fail, the service would not be interrupted and the end users would not experience any problems. In the same way, balancing two or more Web servers with a load balancing machine ensures *Web server high availability*. You can also easily implement *ISS high availability* if you decided to use the ISS function in your architecture (see 4.8.1, “ISS Configuration File” on page 295).

The Web client software must be configured so that all the requests directed to an external Web site are intercepted by a proxy server. The configuration must be issued with the cluster address of the proxy servers, so that the request is first intercepted by the Dispatcher, and then forwarded to the least loaded proxy server. This whole process is completely transparent to the client. In our scenario, the Web site is also clustered and all the requests that come from one of the two proxy servers are in turn intercepted by another load balancing machine, which selects the best available Web server. This will be the Web server to which the request is forwarded.

A great advantage in terms of performance comes from the fact that the Web servers respond directly to the proxy server that submitted the request. The response coming from the Web server does not pass through the load balancing machine that had selected that Web server. In a similar way, the proxy server can respond directly to the client that submitted the request. The response coming from the caching and filtering proxy server goes directly to the client and does not need to pass through the load balancing machine from through which the request had passed. This feature permits the use of a fast network for outbound traffic. Moreover it hides the architecture of the clustered Web site to the proxy level and the architecture of the proxy site to the end users.

As we have explained, the caching and filtering proxy server contacts a Web server only if the requested page is not present in its cache or is up-to-date. As soon as a client requests a cacheable Web page, that page is saved in the cache and served directly from the cache for all subsequent requests.

Notice that two or more caching and filtering proxy servers load-balanced by a Dispatcher machine do not share the same cache. When the Dispatcher forwards a client’s request to one of the clustered proxy servers, that proxy server will have the served Web page stored in its local cache, if such a Web page was cacheable. All the other proxy servers in the same cluster cannot see that Web page, so that if a subsequent request for the same page is forwarded by the Dispatcher to a different caching proxy server, this proxy server will have to retrieve the Web page from the Web site, although this operation was already done by another proxy server in the same cluster.

For this reason, two or more proxy servers load-balanced by a Dispatcher machine do not result in better performances at the very beginning of their activity, since Web pages will likely have to be cached twice. However, recent studies have demonstrated that when the number of users is large, the probability that two load-balanced caching proxy servers have cached the same pages is very high, because multiple users will have requested the same pages several

times and the Dispatcher machine will have distributed the load equally between the clustered proxy servers.

We should also remember that the Caching and Filtering component of IBM WebSphere Performance Pack presents an option to automatically refresh the cache with the most frequently accessed pages. These can be identified by an administrator or determined by a system from the cache logs. Such pages can be retrieved by both the proxy servers at a specific time, when the activities are low (at night, for example), and in this way the probability that all the proxy servers in the same cluster have the same pages in the cache is even higher.

Of course, the approach of load-balancing two caching and filtering proxy servers using the Load Balancing component of IBM WebSphere Performance Pack can result in redundancy of the cached pages. The Cache Array Routing Protocol (CARP) is a specification intended to eliminate this redundancy, but not to eliminate the problems of resulting hot spots for pages in high demand. The Remote Cache Access (RCA) enhances the basic proxy caching component with CARP-like algorithms to eliminate redundant pages but further avoids resultant hot spots through effective combination of these algorithms with load balancing and shared file storage. This feature is available with Distribute Web Traffic Express (DWTE), another IBM offering complementary to IBM WebSphere Performance Pack. However, we did not have the opportunity to use DWTE in our scenarios.

Notice that in this particular scenario we did not implement the filtering function of the Caching and Filtering component of IBM WebSphere Performance Pack. If in your architecture you want to include the filtering function, you can refer to a separate PICS filtering scenario that we built, and that is described in Appendix 3.12, "PICS Scenario" on page 186.

One Note on Caching

According to a study performed at the University of Kaiserslautern and reported in the April 1997 issue of the IEEE Internet Computing, in a real situation, 69.9% of pages are referenced only once. This means that, over a large number of observations, the percentage of requested data that is taken from the proxy cache is nearly 30% of the total requests. The remaining 70% of the data is taken directly from the target Web server (on Internet or intranet). Many external factors can influence the data transferring, such as the bandwidth network, the network traffic and the load on the contacted Web server.

In our scenario, the machines distributed over two token-ring network segments at 16 Mbps:

1. The client-side network segment was composed by the client machine and the two clustered proxy servers balanced by a load balancing machine.
2. The server-side network segment was composed by the three clustered Web servers balanced by another load balanced machine.

These two LAN segments were separated by a RISC/6000 machine provided by two network interfaces, playing the role of an IP router. All the details on how we

built such an environment are described in Appendix A, “Network Configuration of the Scenario Environments” on page 405.

Our purpose in this chapter is to show you how you can benefit from a regional and local access ISP architecture. The advantages of having a clustered Web site, where the Web servers share the same contents through an AFS file system, have already been shown in Chapter 5, “Enterprise Campus Scenario” on page 349. In this chapter we want to show you the advantages of load-balancing multiple caching and filtering proxy servers. For this reason we built this scenario in two steps:

1. First we implemented a *regional and local access ISP basic scenario*, where the architecture is very similar to Figure 309 on page 367, except that the proxy function was implemented by a single caching proxy server, with no Dispatchers.
2. Then we enhanced the basic scenario and built a *complete regional and local access ISP scenario*, involving two caching and proxy servers, load-balanced by a Dispatcher machine, as shown in Figure 309 on page 367.

In both the steps we used WebStone to generate the workload (see Appendix B, “Scenario Basic Configuration” on page 411). We also extracted the output information generated by WebStone to measure the Web site performances in both scenarios and see how performances increase when using a Dispatcher machine to balance two caching proxy servers. Notice that the results of our measurements can be considered valid only keeping in mind the limitations of the environment we were working on. As we discuss in the box “Value of Performance Measurements” on page 417, when we built our scenarios, our main purpose was *not* to measure the performances of the Web sites we had created. Our scenarios were aimed at demonstrating how the various components of IBM WebSphere Performance Pack can be combined together to provide complex and powerful architectures. We implemented such architectures and we tried them simulating a workload with WebStone.

Performance measurements in an environment as limited as ours do not have a real value. We could have used more powerful machines and a larger number of clients, tuned our machines to get a better utilization of CPU and a faster disk access, used a faster network, etc. However, keeping in mind these limitations, we think it is interesting to see how the performances of a certain Web site, although not optimized, increase when using IBM WebSphere Performance Pack.

6.1 First Step: Regional and Local Access ISP Basic Scenario

We implemented this scenario as a first step before building the complete enterprise campus scenario. This scenario has a value also in terms of performances, because we compare the performances of this architecture with the performances measured while using two clustered caching proxy servers load balanced by a Dispatcher machine.

The following diagram offers a graphical representation of the application flow:

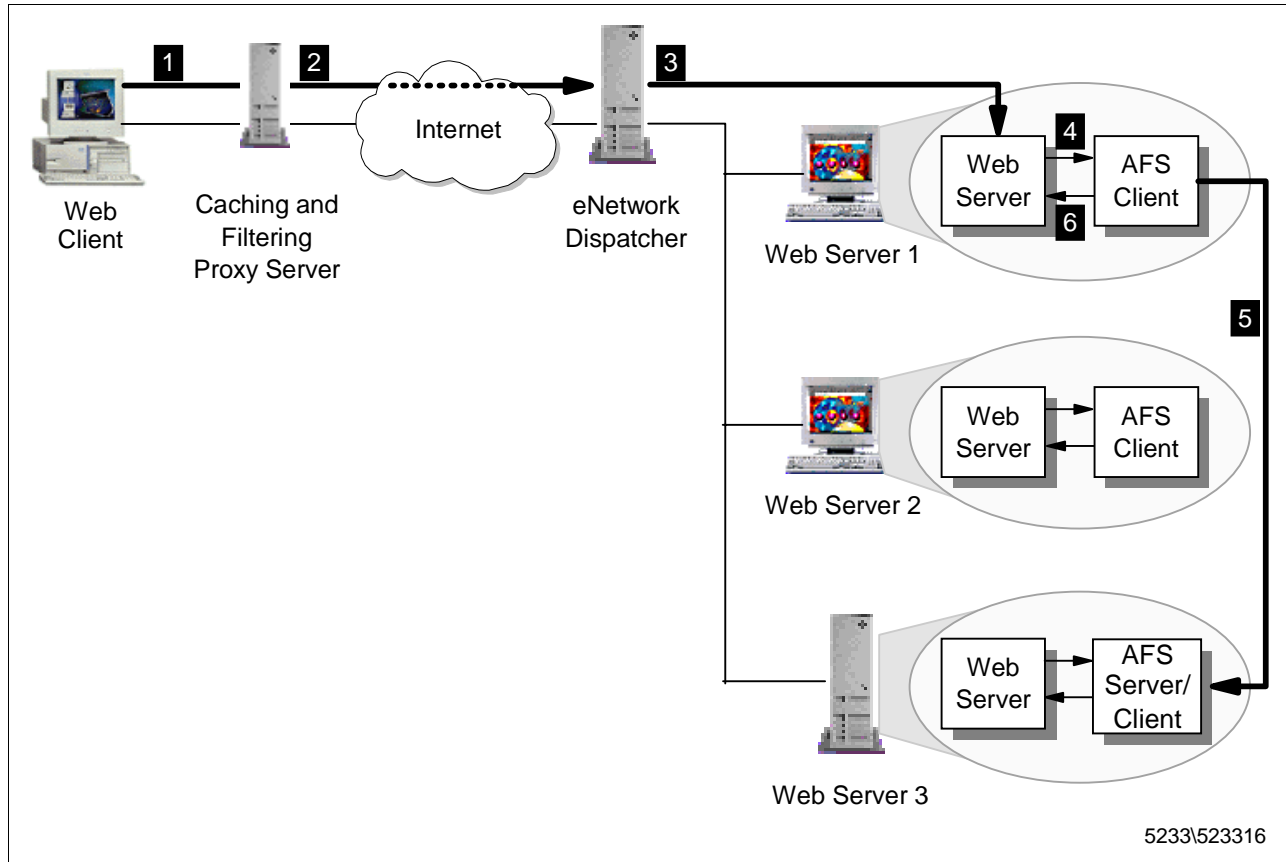


Figure 310. Flow Diagram of the Regional and Local Access ISP Basic Scenario

The flow of the regional and local access basic scenario that we built in our scenario is described in the following list:

1. The Web client software issues a request using the cluster address of the Web site (for instance `http://9.24.104.105/afs/webstone/html/file2k.html`). Since the browser is configured to submit its requests to a caching proxy server, the proxy server intercepts the client's request.
2. The caching proxy server serves the requested Web page directly if this is in its local cache, otherwise the request is forwarded to the Web site. Since the IP address specified in the URL was the cluster address, it is the Dispatcher that receives the request.
3. The Dispatcher selects the appropriate Web server (in this example, Web Server 1) and forwards the request to it.
4. The `httpd` daemon running on Web Server 1 interprets the HTTP request and issues a local I/O call that is intercepted by the AFS Cache Manager to retrieve the data.
5. The Cache Manager retrieves the data from its local cache if the requested data is present and up-to-date, or requests it from the AFS server.
6. Upon retrieving data, the Cache Manager gives it back to the `httpd` daemon, which is able to honor the client's request.

Notice that the response from the Web server flows directly to the proxy server that had submitted the request without passing through the Dispatcher of the clustered Web site.

The hardware, software and network architecture of the regional and local access ISP scenario we implemented is shown in the following table:

Table 14. Basic Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	aixncf157	172.16.1.5	AIX 4.3.1	Web Client
IBM RS/6000 43P	venus	172.16.1.3	AIX 4.3.1	Caching Proxy Server
IBM RS/6000 43P	rs600012	I) 172.16.1.1 E) 9.24.104.124	AIX 4.3.1	IP Router
IBM RS/6000 43P	1) rs600023 2) clusterend	1) 9.24.104.128 2) 9.24.104.105	AIX 4.3.1	Load Balancing
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3.1	AFS Server and Client, Web Server
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	AFS Client, Web Server
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	AFS Client, Web Server

The role of the Web client was provided by WebStone 1.1, while the Web server functionality was provided by Lotus Domino Go Webserver 4.6.2.5. All the details on the configuration of the environment shown in the above table can be found in Appendix A, "Network Configuration of the Scenario Environments" on page 405. The configuration of WebStone can be found in Appendix B, "Scenario Basic Configuration" on page 411.

As we explained in Appendix B.3.3, "WebStone Configuration Files" on page 413, the workload was light dynamic, meaning it was made up of a mix of static and dynamic pages.

We defined a cluster of three servers on TCP port 80, since we wanted the Dispatcher to load balance only HTTP requests. The details of the Dispatcher configuration we implemented are shown in Figure 297 on page 358, Figure 321 on page 384, Figure 322 on page 385 and Figure 323 on page 386.

Notice that this scenario is an enhancement from the scenario described in 5.2, "Second Step: Enterprise Campus Complete Scenario" on page 355. So you can safely repeat the same configuration steps, the only difference being that this time you should put, in the client-side, a caching proxy server to which the clients requests are directed. The caching proxy server should have the caching function enabled. There are no differences in the server-side configuration.

These are the configuration steps in detail:

1. Set up all the machines (see Appendix A., "Network Configuration of the Scenario Environments" on page 405 and Appendix B., "Scenario Basic Configuration" on page 411).

2. Install and configure the caching proxy server to which the client's requests are directed (see 3.7, "Installation of the Caching and Filtering Component" on page 120 and 3.8, "Basic Configuration" on page 149).

We did not use the PICS filtering function in this architecture, but if you want to implement it in your environment, you can refer to 3.12, "PICS Scenario" on page 186.

3. Install and configure the load balancing component (see 4.5, "Installation of the Load Balancing Component" on page 229 and 4.6.3, "Dispatcher Configuration" on page 248).

We did not use the ISS function in this architecture, but if you want to implement ISS in your environment you can refer to 4.8, "Load Balancing Scenario Using the Dispatcher and ISS Functions" on page 294.

4. Configure the Web servers to be part of the clustered Web site (see 4.6.4, "TCP Servers Configuration" on page 267).
5. Install and configure the File Sharing component on the Web server machines (see 2.4, "Installation and Configuration of the File Sharing Component" on page 21).

As we explained in Appendix B.3.3, "WebStone Configuration Files" on page 413, the workload generated by the client machine in our scenario was light dynamic, meaning it was made up of a mix of static and dynamic pages.

This is the WebStone testbed file we used:

```
#Testbed file used for generating the load to the caching proxy server
#addressing the Dispatcher plus three Web Servers
ITERATIONS="1"
MINCLIENTS="20"
MAXCLIENTS="100"
CLIENTINCR="10"
TIMEPERRUN="2"
SERVER="venus"
PORTNO=80
WEBSTONEROOT="/home/guest/WebStone-1.1"
CLIENTS="aixncf157"
CLIENTACCOUNT=root
CLIENTPASSWORD=pistoia
TMPDIR=/tmp
RCP="rcp"
RSH="rsh"
```

Figure 311. testbed File

Notice that the `SERVER` parameter is set to `venus`, which was the host name associated to the IP address of the caching proxy server machine in our environment.

The WebStone filelist file for light dynamic workload is shown in the following figure:

```

#Modified Silicon Surf model; added light dynamic content use
8
40 2
http://clusterend/afs/webstone/html/file2k.html
http://clusterend/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072
25 2
http://clusterend/afs/webstone/html/file1k.html
http://clusterend/afs/webstone/html/file5k.html
15 2
http://clusterend/afs/webstone/html/file4k.html
http://clusterend/afs/webstone/html/file6k.html
5 1
http://clusterend/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=7168
4 4
http://clusterend/afs/webstone/html/file8k.html
http://clusterend/afs/webstone/html/file9k.html
http://clusterend/afs/webstone/html/file10k.html
http://clusterend/afs/webstone/html/file11k.html
4 5
http://clusterend/afs/webstone/html/file12k.html
http://clusterend/afs/webstone/html/file14k.html
http://clusterend/afs/webstone/html/file15k.html
http://clusterend/afs/webstone/html/file17k.html
http://clusterend/afs/webstone/html/file18k.html
6 1
http://clusterend/afs/webstone/html/file33k.html
1 1
http://clusterend/afs/webstone/html/file200k.html

```

Figure 312. WebStone filelist File

If you compare the above filelist with the filelist shown in Figure 302 on page 362, you can see that when requests to a Web site must pass through a proxy, the filelist file must specify the fully qualified URL for each Web page it is going to retrieve. You can also see that we indicated the cluster address of the Web site, so that all incoming requests would be intercepted by the Dispatcher machine.

After launching WebStone, we captured the following screens from the Server Monitor of the Dispatcher, while the WebStone was generating a workload of 70 concurrent clients:

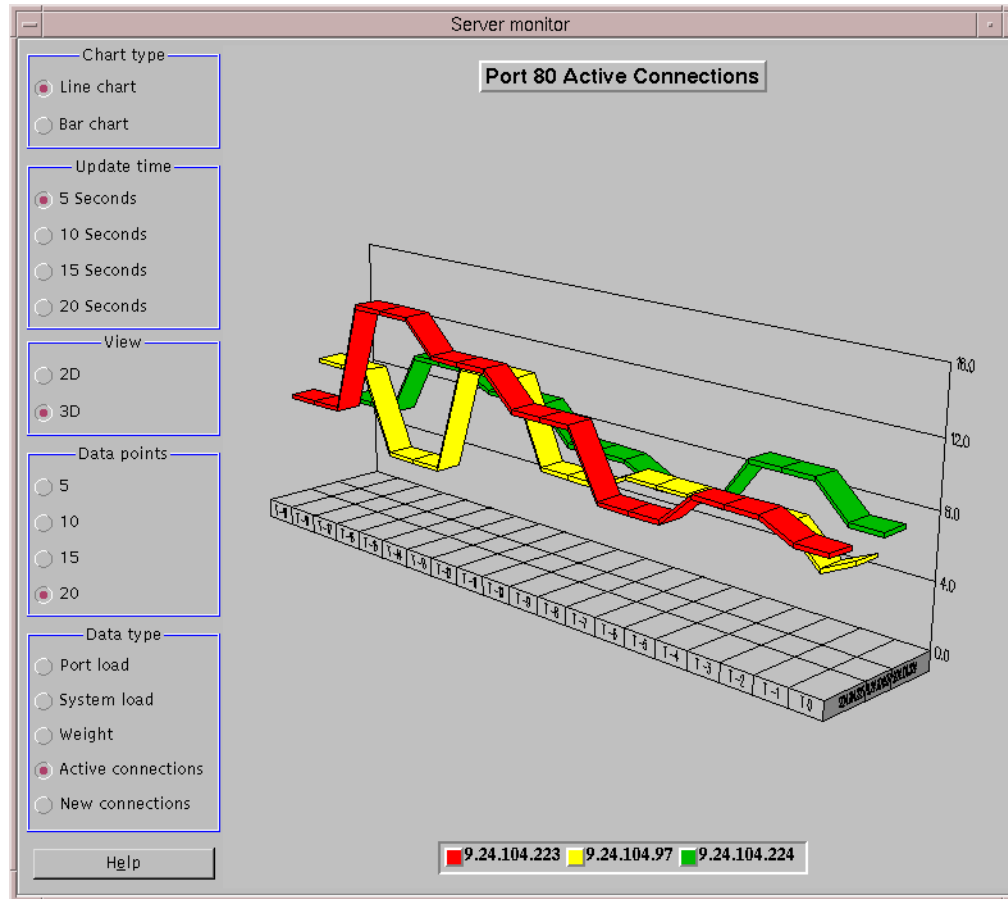


Figure 313. Active Connections 70 Clients

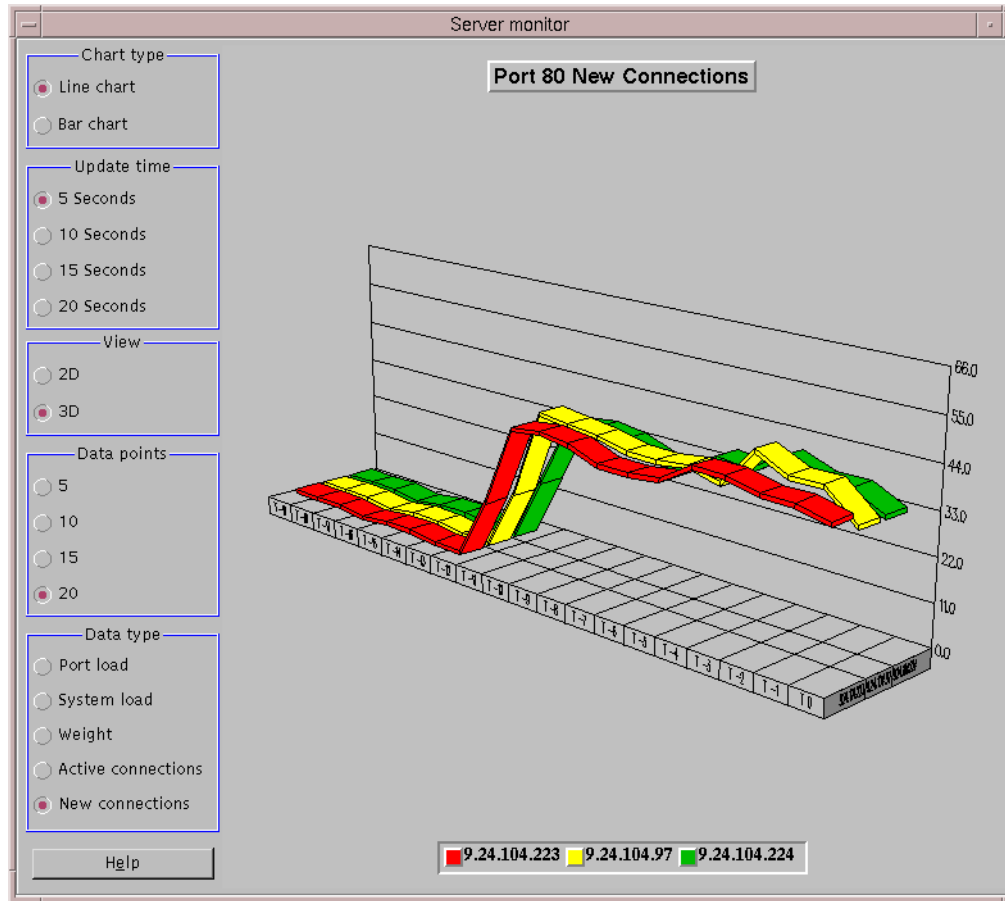


Figure 314. New Connections 70 Clients

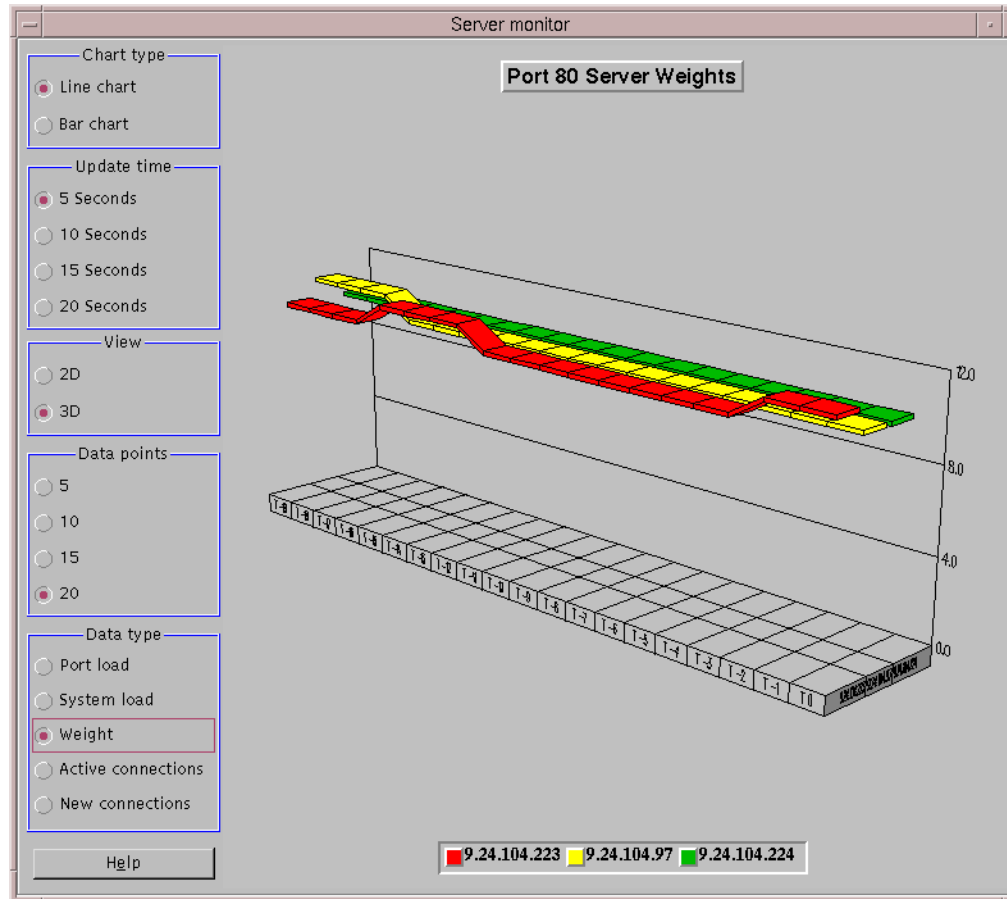


Figure 315. Servers' Weights 70 Clients

The above diagrams show the activity of the three Web servers during an interval of intense workload. They also demonstrate how the workload is distributed by the Dispatcher machine among all the available servers. The workload represented in the above diagrams was completely generated by dynamic requests, which were achieved by HTML pages invoking CGI-Bin programs. We verified that all the static HTML pages were already cached by the caching proxy server, which was able to serve them from its own local cache. This means that workload would have been even higher, had we not used a caching proxy server.

After running WebStone, we got the following performance outputs:

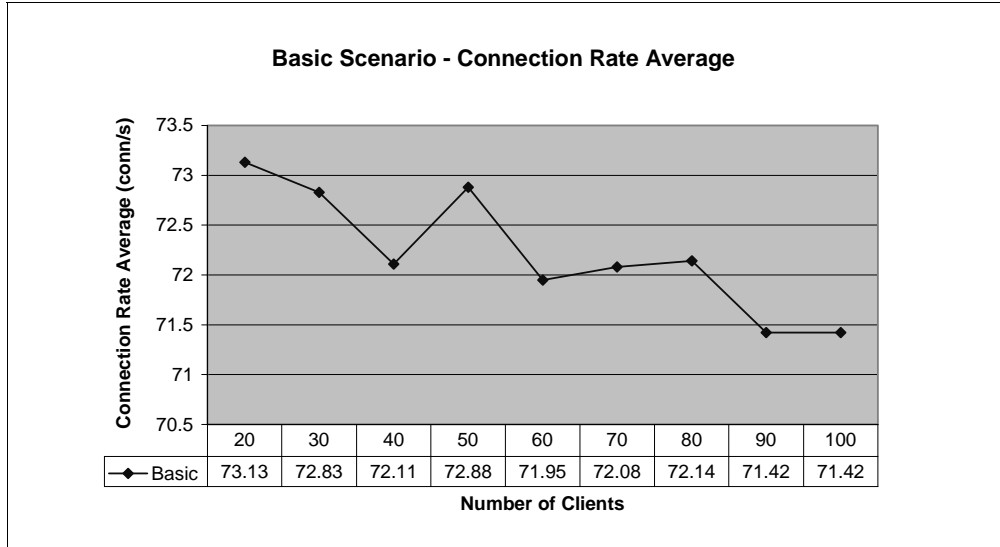


Figure 316. Basic Scenario - Connection Rate Average

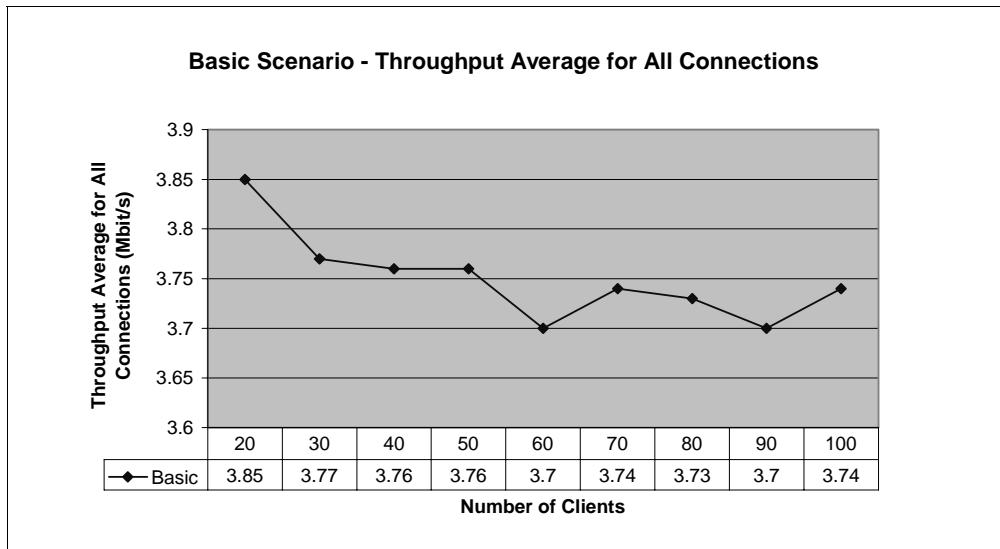


Figure 317. Basic Scenario - Throughput Average for All Connections

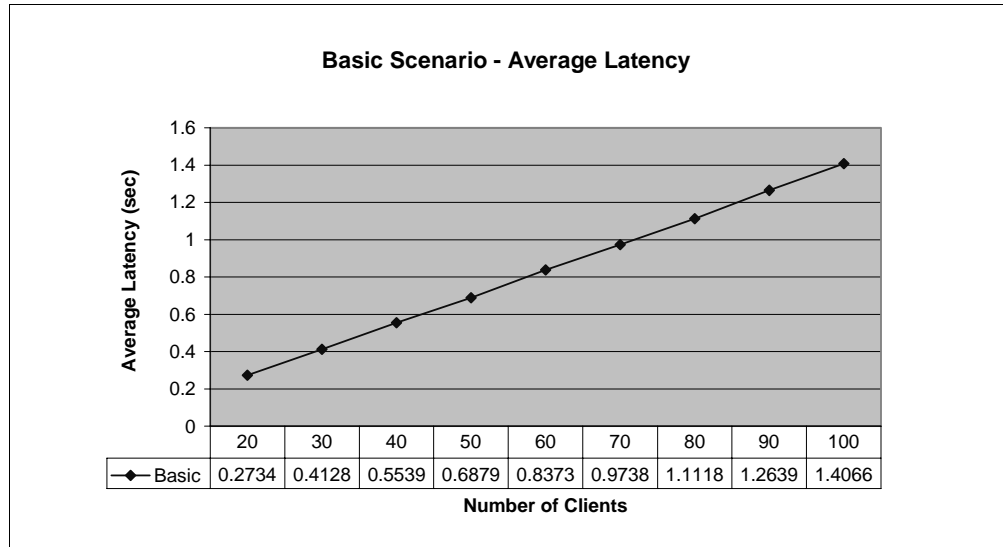


Figure 318. Basic Scenario- Average Latency

The above diagrams demonstrate that performances increase dramatically when using a caching proxy server. In our environment, even when 100 concurrent clients were simultaneously submitting HTTP requests to the Web site, the average client rate for data to be returned is only 1.4 seconds. We recommend that you compare these results with those shown in 5.2, “Second Step: Enterprise Campus Complete Scenario” on page 355.

6.2 Second Step: Regional and Local Access ISP Complete Scenario

In this section we show you how to enhance the basic scenario shown in 6.1, “First Step: Regional and Local Access ISP Basic Scenario” on page 370, in order to build a regional and local access ISP complete scenario.

The flow of the regional and local access ISP complete scenario we built in our environment is represented in the following diagram:

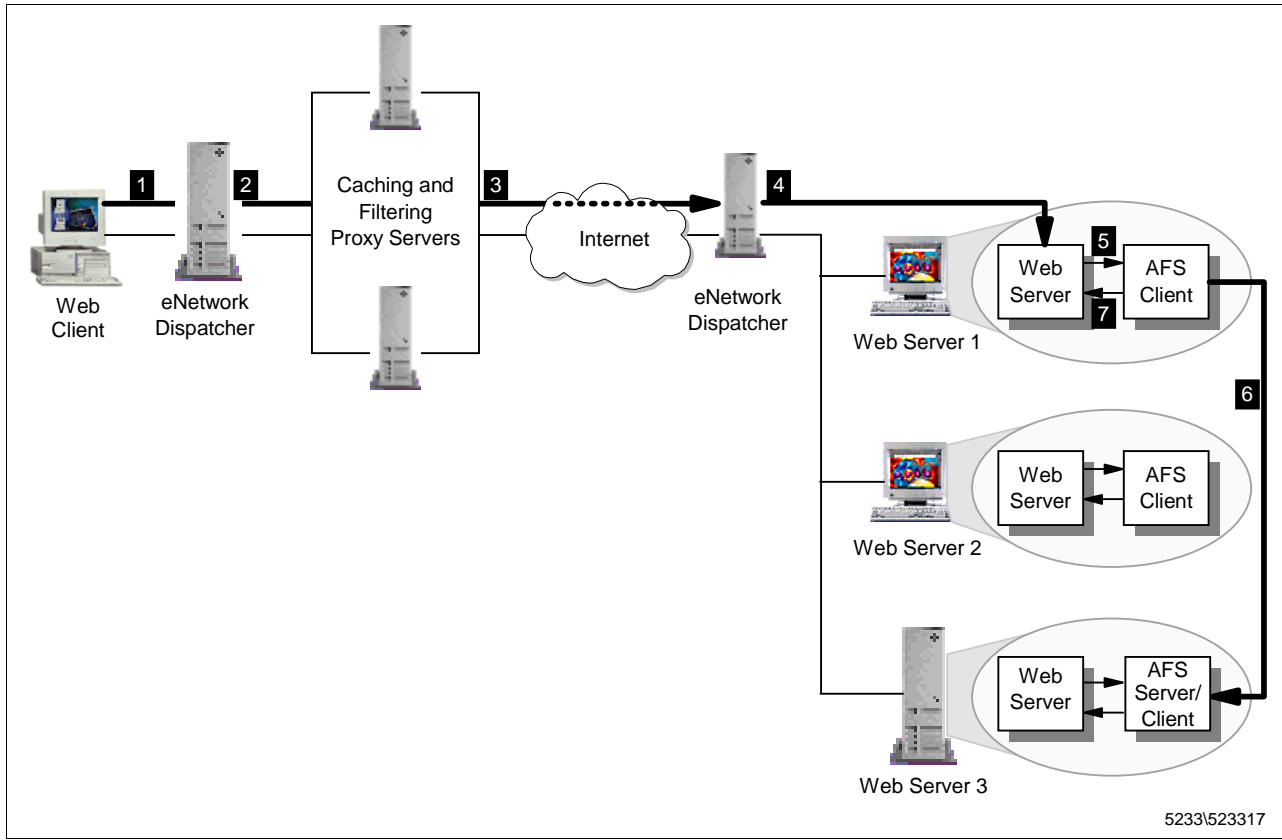


Figure 319. Regional and Local Access ISP Flow Diagram

As you can see from the above diagram, two clustered sites are now involved:

1. A caching proxy server clustered site, where a Dispatcher machine intercepts the client's request, selects the appropriate caching proxy server and forwards the request to that caching proxy server
2. A Web server clustered site, where a Dispatcher machine intercepts the client's request forwarded by the caching proxy server clustered site, selects the appropriate Web server and forwards the request to the that Web server

This architecture ensures both Web server and caching proxy server high availability, plus ISS high availability if you decide to use the ISS function in your architecture (see 4.8.1, "ISS Configuration File" on page 295).

The following list gives more details on the application flow graphically represented in the above diagram:

1. The Web client software issues a request using the cluster address of the Web site (for instance `http://9.24.105/afs/webstone/html/file2k.html`). Since the browser is configured to submit its requests to the cluster address of a caching proxy server site, the client's request is intercepted by the Dispatcher machine.
2. The Dispatcher selects the appropriate caching proxy server in the cluster and forwards the request to it.
3. The caching proxy server selected by the Dispatcher serves the requested Web page directly if this is in its local cache, otherwise the request is

forwarded to the Web site. Since the IP address specified in the URL was the cluster address of the Web site, it is the Dispatcher that receives the request.

4. The Dispatcher selects the appropriate Web Server (in this example, Web Server 1) and forwards the request to it.
5. The httpd daemon running on Web Server 1 interprets the HTTP request and issues a local I/O call that is intercepted by the AFS Cache Manager to retrieve the data.
6. The Cache Manager retrieves the data from its local cache if the requested data is present and up-to-date, or requests it from the AFS server.
7. Upon retrieving data, the Cache Manager gives it back to the httpd daemon that is able to honor the client's request.

Notice that the response from the Web server flows directly to the proxy server that submitted the request without passing through the Dispatcher of the clustered Web site. In the same way, the response from the proxy server flows directly to the client without passing through the Dispatcher of the clustered proxy server site.

The hardware, software and network configuration of the regional and local access ISP scenario we implemented is shown in the following table:

Table 15. Complete Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	aixncf157	172.16.1.5	AIX 4.3.1	WebStone
IBM PC 365	1) wtr05219 2) clusternt	1) 172.16.1.4 2) 172.16.1.6	Windows NT Server 4.0	Proxy Server Load Balancing
IBM RS/6000 43P	rs600022	172.16.1.2	AIX 4.3.1	Caching Proxy Server
IBM RS/6000 43P	venus	172.16.1.3	AIX 4.3.1	Caching Proxy Server
IBM RS/6000 43P	rs600012	I) 172.16.1.1 E) 9.24.104.124	AIX 4.3.1	IP Router
IBM RS/6000 43P	1) rs600023 2) clusterend	1) 9.24.104.128 2) 9.24.104.105	AIX 4.3.1	Web Server Load Balancing
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3.1	AFS Server and Client, Web Server
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	AFS Client, Web Server
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	AFS Client, Web Server

The role of the Web client was provided by WebStone 1.1, while the Web server functionality was provided by Lotus Domino Go Webserver 4.6.2.5. All the details on the configuration of the environment shown in the above table can be found in Appendix A, "Network Configuration of the Scenario Environments" on page 405. The configuration of WebStone can be found in Appendix B, "Scenario Basic Configuration" on page 411.

As we explained in Appendix B.3.3, “WebStone Configuration Files” on page 413, the workload was light dynamic, meaning it was made up of a mix of static and dynamic pages.

We defined a cluster of three servers on the TCP port 80, since we wanted the Dispatcher to load balance only HTTP requests.

Notice that this scenario is an enhancement from the scenario described in 6.1, “First Step: Regional and Local Access ISP Basic Scenario” on page 370. So you can safely repeat the same configuration steps, the only difference being that this time you should put in the client-side two caching proxy servers load balanced by a Dispatcher machine, to which the clients’ requests are directed. To improve performance, both the caching proxy servers should have the caching function enabled. There are no differences in the server-side configuration.

These are the configuration steps in detail:

1. Set up all the machines (see Appendix A, “Network Configuration of the Scenario Environments” on page 405 and Appendix B, “Scenario Basic Configuration” on page 411).
2. Install and configure the two Dispatcher machines on the client and server side (see 4.5, “Installation of the Load Balancing Component” on page 229 and 4.6.3, “Dispatcher Configuration” on page 248).

We did not use the ISS function in this architecture, but if you want to implement ISS in your environment you can refer to 4.8, “Load Balancing Scenario Using the Dispatcher and ISS Functions” on page 294.

3. Install and configure the two (or more) caching proxy servers (see 3.7, “Installation of the Caching and Filtering Component” on page 120 and 3.8, “Basic Configuration” on page 149).

We did not use the PICS filtering function in this architecture, but if you want to implement it in your environment, you can refer to 3.12, “PICS Scenario” on page 186.

4. Configure the Web servers to be part of the clustered Web site and the caching proxy servers to be part of the clustered proxy server site (see 4.6.4, “TCP Servers Configuration” on page 267).
5. Install and configure the File Sharing component on the Web server machines (see 2.4, “Installation and Configuration of the File Sharing Component” on page 21).

About the load balancing configuration of the caching proxy server site, we had defined a cluster of two TCP servers on the TCP port 80, since we wanted our Dispatcher to load-balance only HTTP requests against the two caching proxy servers.

We created an alias on the network interface tr0 of the Dispatcher machine to the cluster address 172.16.1.6 of the caching proxy server site (see 4.6.3.6, “Aliasing the Network Interface to the Cluster Address” on page 255). Then, on both caching proxy servers belonging to the cluster, we aliased the loopback device lo0 to the cluster address (see 4.6.4.1, “Aliasing the Loopback Device on AIX” on page 267 and 4.6.4.2, “Aliasing the Loopback Device on Windows NT” on page 268).

The configuration of the Dispatcher that distributed the load between the two caching proxy servers is shown in the following four figures:

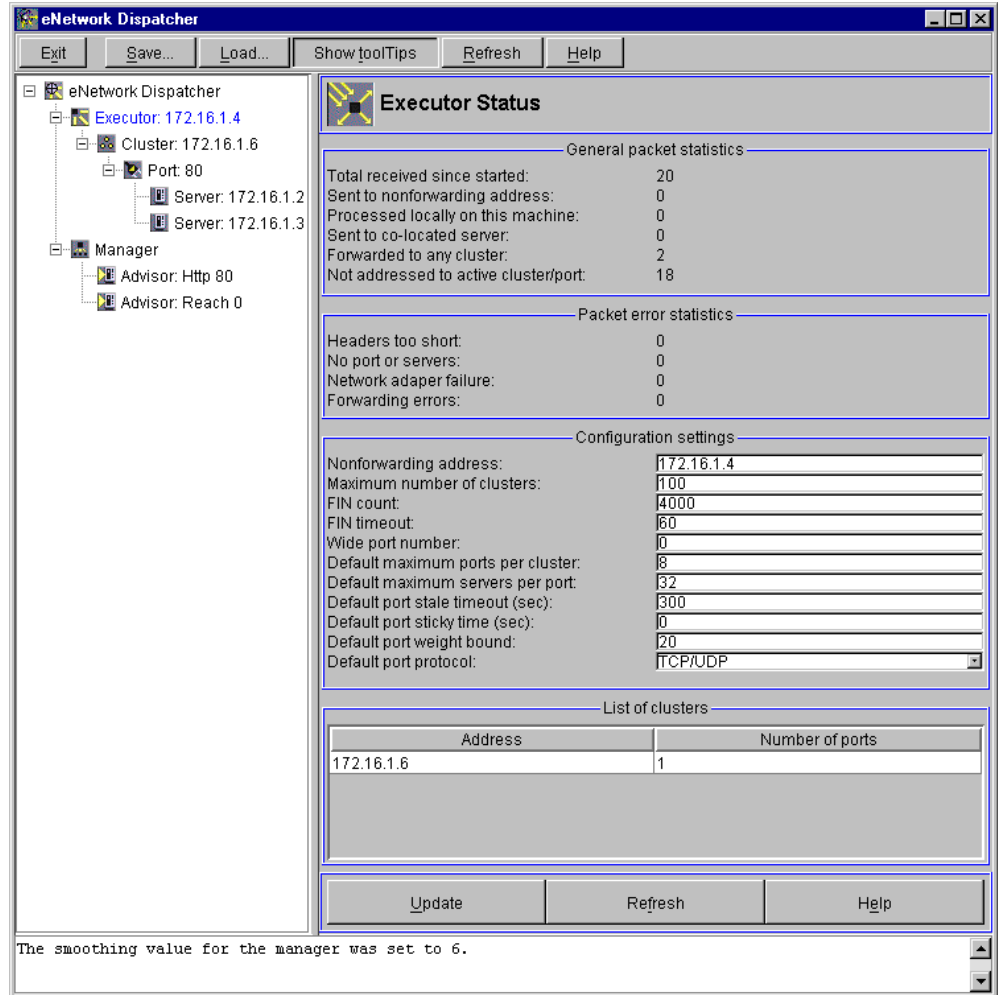


Figure 320. Proxy Dispatcher Configuration - Executor Status

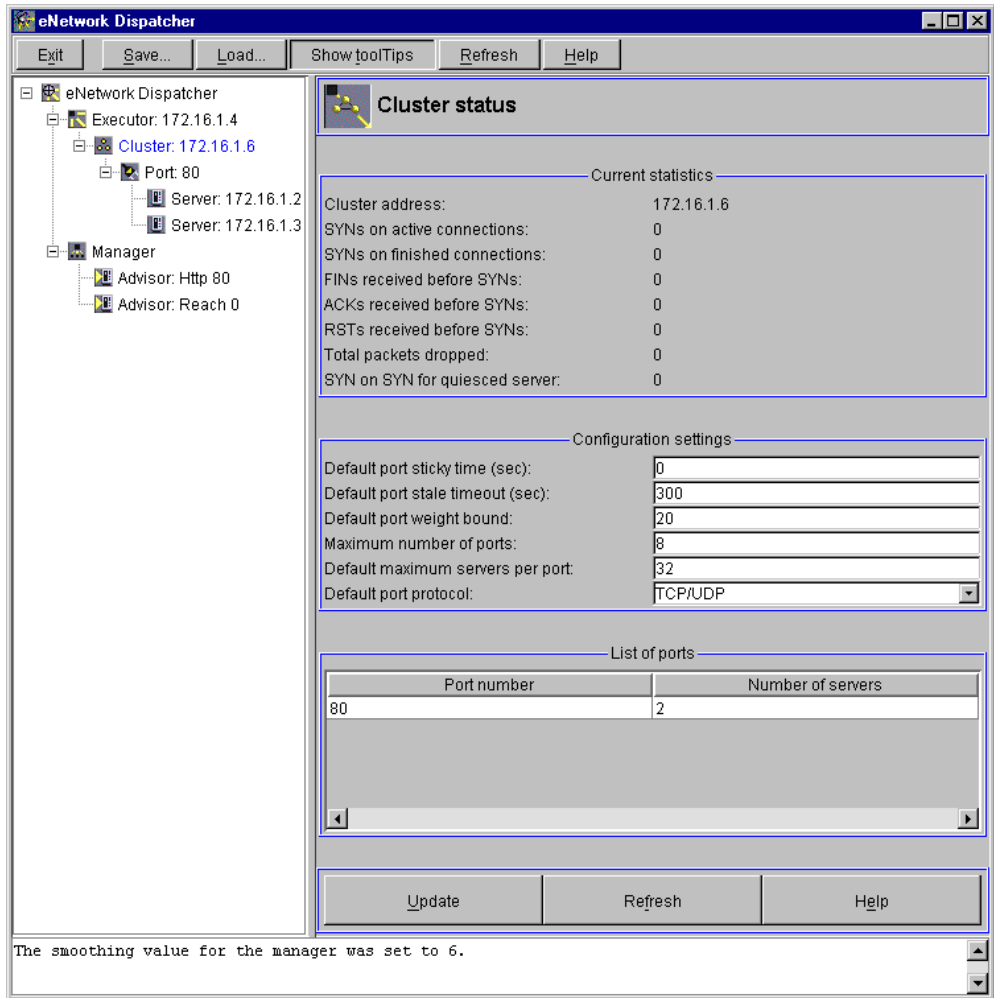


Figure 321. Proxy Dispatcher Configuration - Cluster Status

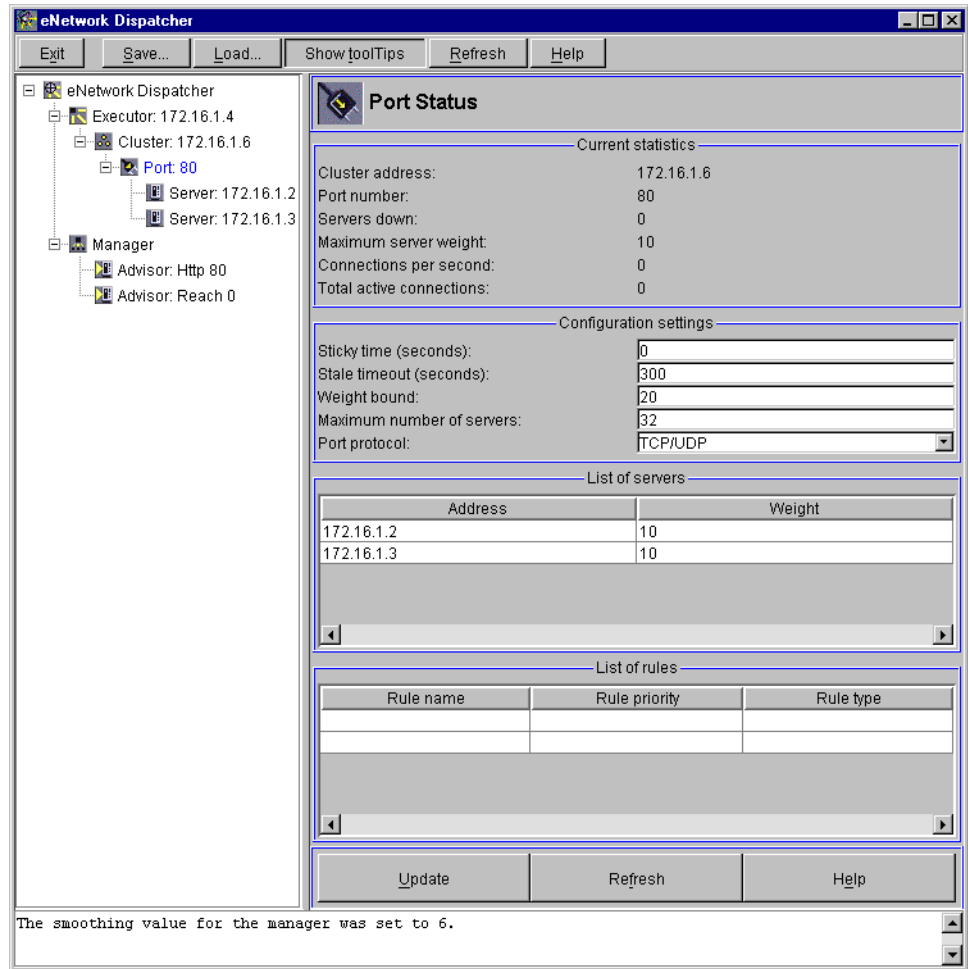


Figure 322. Proxy Dispatcher Configuration - Port Status

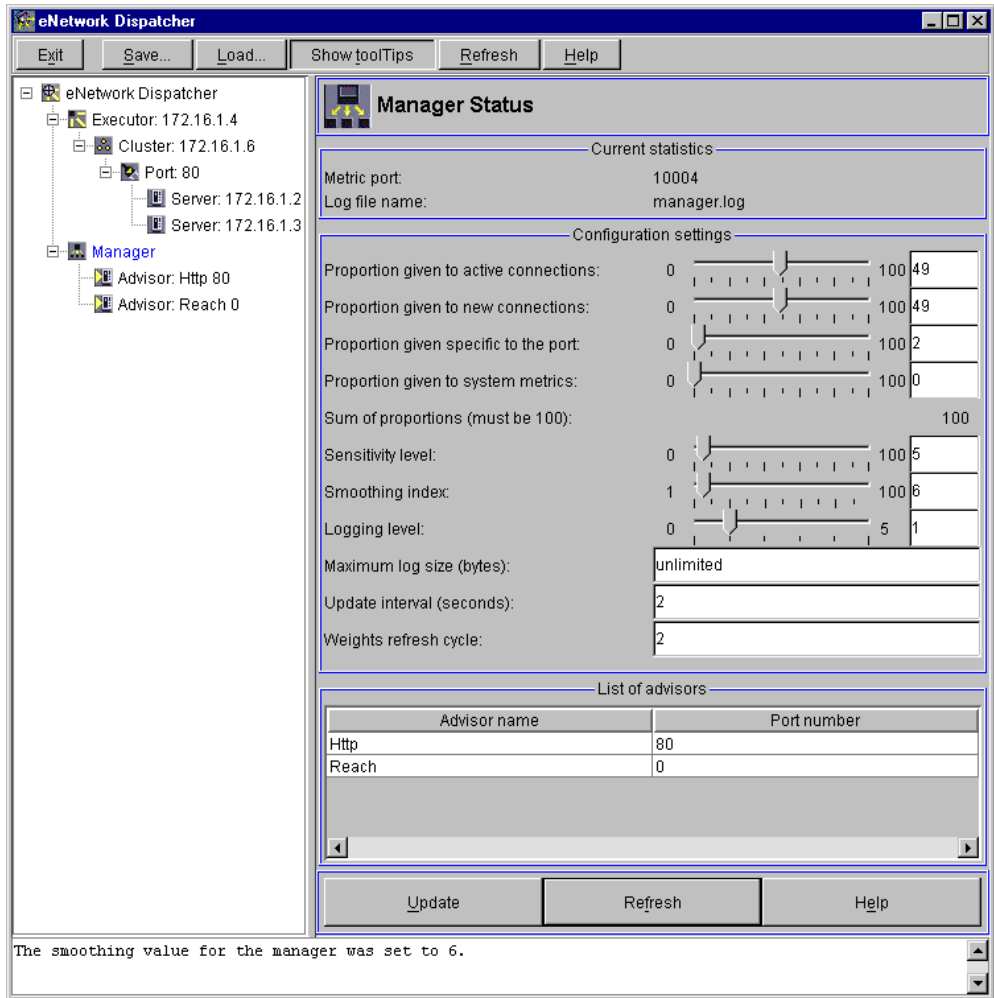


Figure 323. Proxy Dispatcher Configuration - Manager Status

The details of the Dispatcher configuration we implemented in the clustered Web site are shown in Figure 297 on page 358, Figure 321 on page 384, Figure 322 on page 385 and Figure 323 on page 386.

As we explained in B.3.3, “WebStone Configuration Files” on page 413, the workload generated by the client machine in our scenario was light dynamic, meaning it was made up of a mix of static and dynamic pages.

This is the WebStone testbed file we used:


```
#Testbed file used for generating the load to the two caching proxy servers
#load balanced by the Dispatcher with cluster address clusternt,
#addressing the Dispatcher with cluster address clusterend plus three Web
#Servers
ITERATIONS="1"
MINCLIENTS="20"
MAXCLIENTS="100"
CLIENTINCR="10"
TIMEPERRUN="2"
SERVER="clusternt"
PORTNO=80
WEBSTONEROOT="/home/guest/WebStone-1.1"
CLIENTS="aixncf157"
CLIENTACCOUNT=root
CLIENTPASSWORD=pistoia
TMPDIR=/tmp
RCP="rcp"
RSH="rsh"
```

Figure 324. WebStone testbed File

Notice that the `SERVER` parameter was set to `clusternt`, which was the host name associated to the cluster address of the caching proxy server clustered site.

The WebStone filelist file for light dynamic workload is shown in the following figure:

```

#Modified Silicon Surf model; added light dynamic content use
8
40 2
http://clusterend/afs/webstone/html/file2k.html
http://clusterend/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072
25 2
http://clusterend/afs/webstone/html/file1k.html
http://clusterend/afs/webstone/html/file5k.html
15 2
http://clusterend/afs/webstone/html/file4k.html
http://clusterend/afs/webstone/html/file6k.html
5 1
http://clusterend/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=7168
4 4
http://clusterend/afs/webstone/html/file8k.html
http://clusterend/afs/webstone/html/file9k.html
http://clusterend/afs/webstone/html/file10k.html
http://clusterend/afs/webstone/html/file11k.html
4 5
http://clusterend/afs/webstone/html/file12k.html
http://clusterend/afs/webstone/html/file14k.html
http://clusterend/afs/webstone/html/file15k.html
http://clusterend/afs/webstone/html/file17k.html
http://clusterend/afs/webstone/html/file18k.html
6 1
http://clusterend/afs/webstone/html/file33k.html
1 1
http://clusterend/afs/webstone/html/file200k.html

```

Figure 325. WebStone filelist File

When requests pass through a proxy, notice that the filelist file specifies the fully qualified URL for each Web page it is going to retrieve. You can also see that we indicated the cluster address of the Web site, so that all incoming requests would be intercepted by the Dispatcher machine.

After launching WebStone, we captured the following screens from the Server Monitor of the Dispatcher on the clustered proxy server site, while the WebStone was generating a workload of 100 concurrent clients:

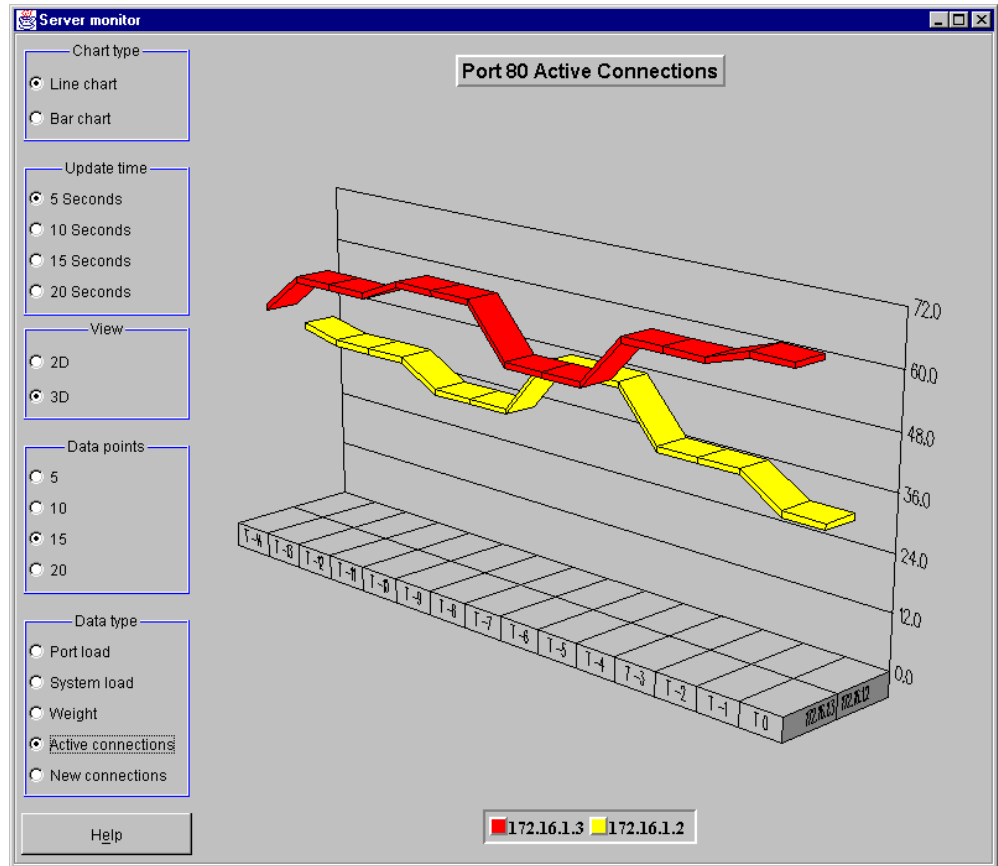


Figure 326. Active Connections 100 Clients

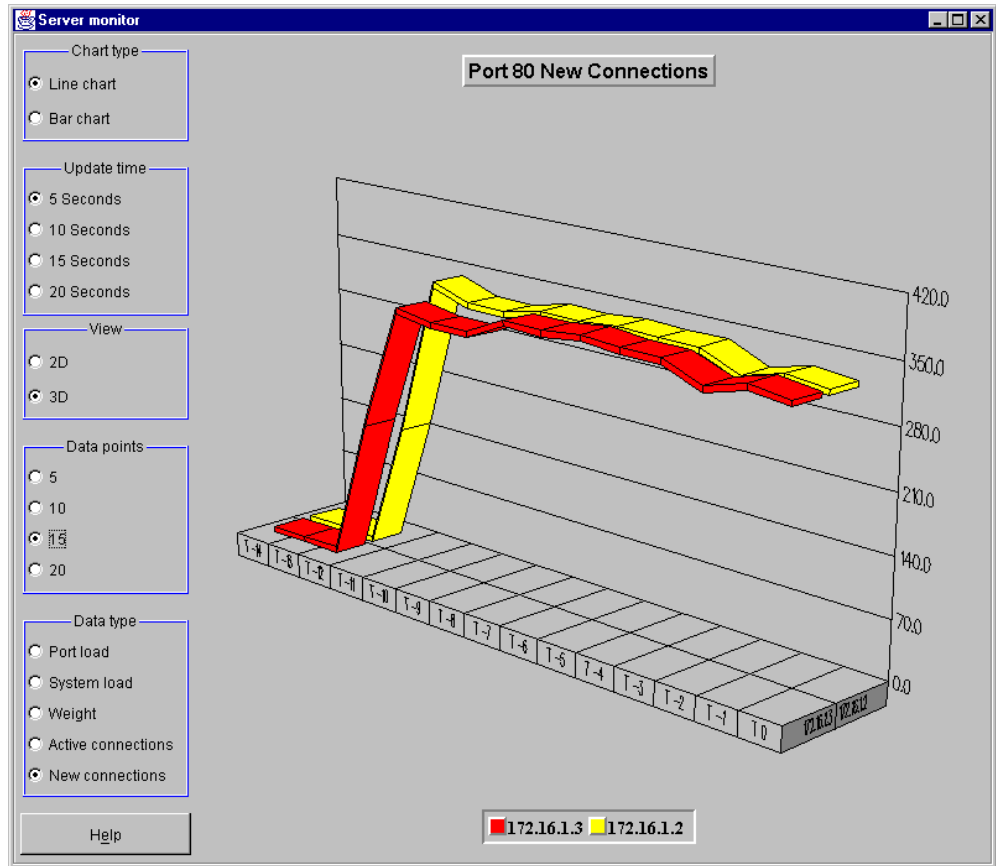


Figure 327. New Connections 100 Clients

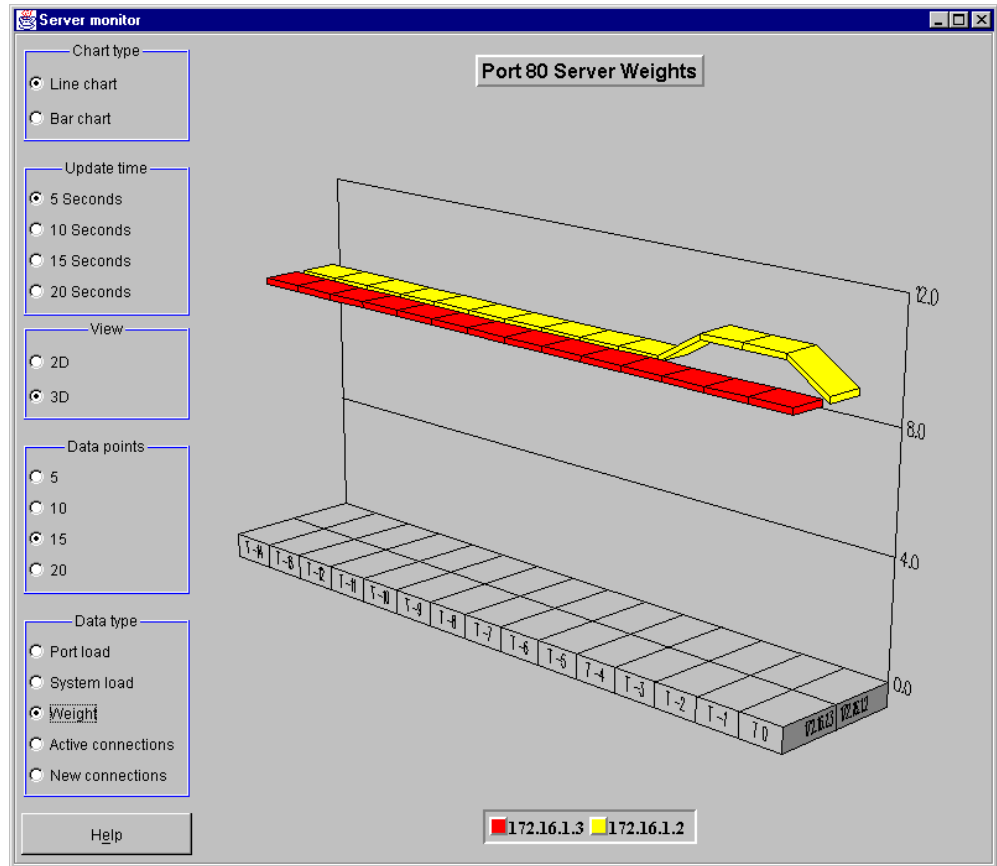


Figure 328. Servers' Weights 100 Clients

The above diagrams represent the activity of the two caching proxy servers during an interval of intense workload. They also demonstrate how the workload is distributed by the Dispatcher machine among all the available proxy servers. The workload represented in the above diagrams was generated:

- 25% by dynamic requests, which were achieved by HTML pages invoking CGI-Bin programs. The caching proxy servers cannot cache dynamic pages, so what a caching proxy server does when a request for a dynamic page arrives is to forward it directly to the Web site.
- 75% by static requests for regular and cacheable HTML pages. A caching proxy server checks if a requested static page is already in its own cache. If yes, the page is served directly to the client that requested it. If not, the request is forwarded to the Web site, and when the Web site sends the response back to the proxy server, the proxy caches the Web page and serves it back to the client that requested it.

We verified that all the static HTML pages were being regularly cached by the caching proxy servers, which then were able to serve them from their own local cache.

During our experience, we also captured the following screens from the System Monitor of the Dispatcher cluster end that load balanced the three Web servers. Also in this case the screen captures refer to a workload of 100 concurrent clients simultaneously submitting their requests:

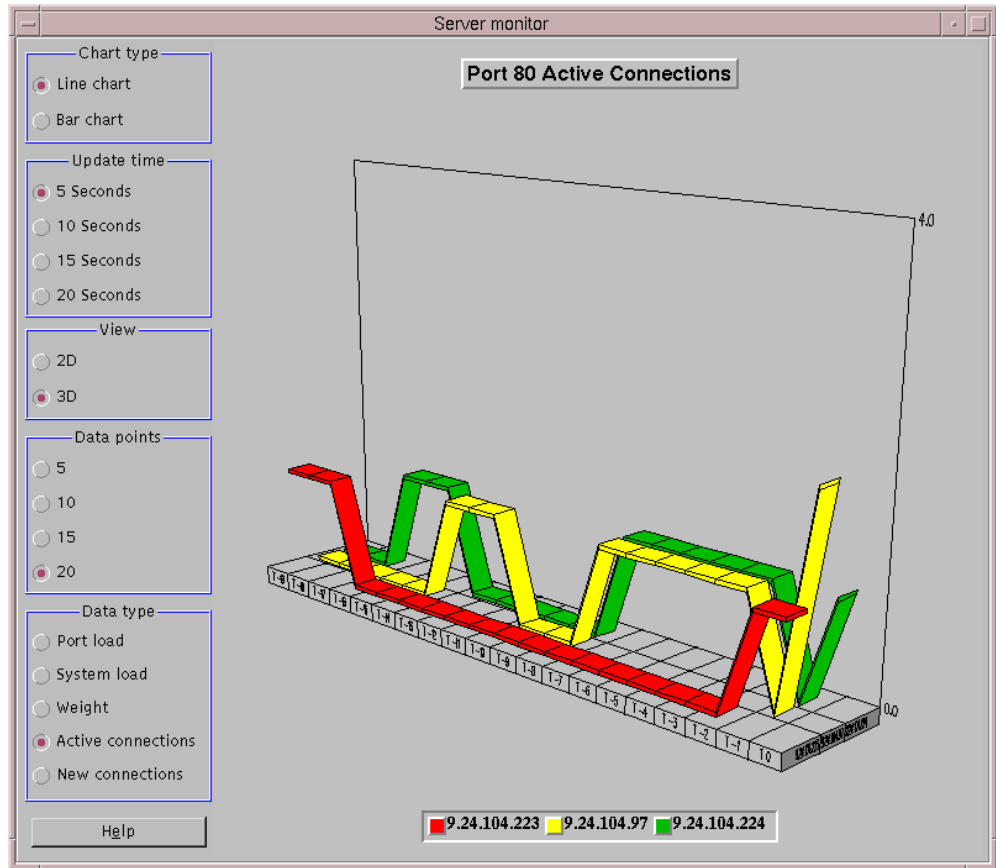


Figure 329. Active Connection's 100 Clients

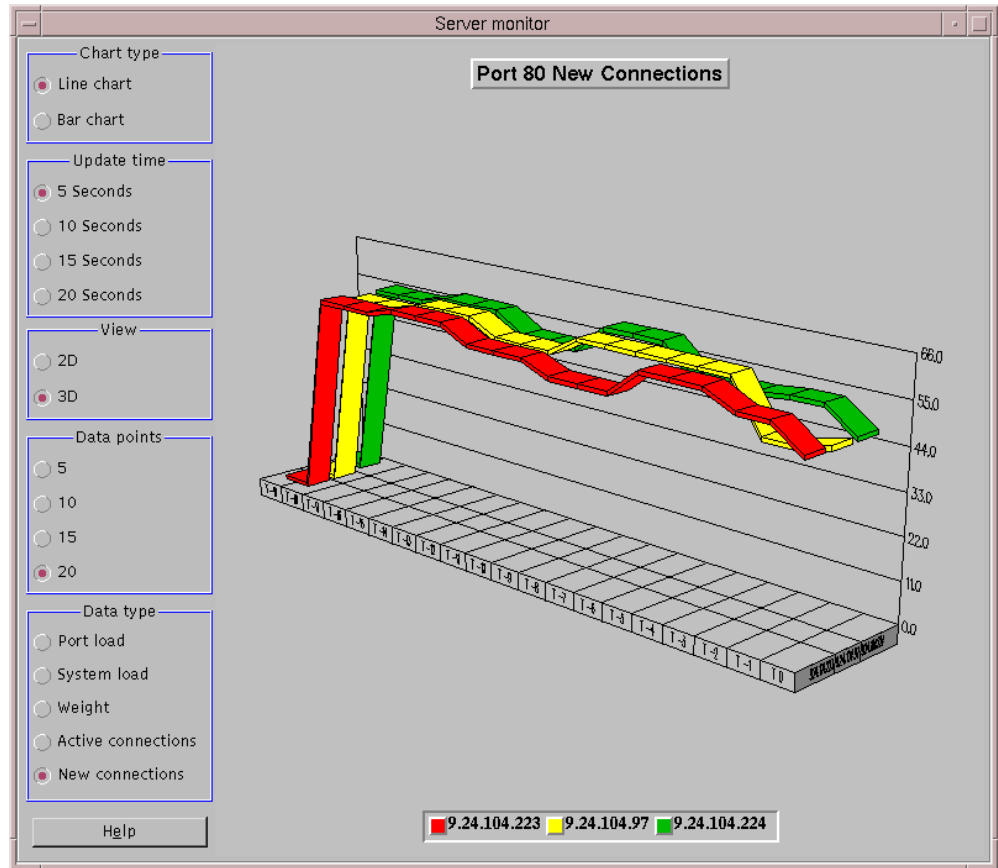


Figure 330. New Connection's 100 Clients

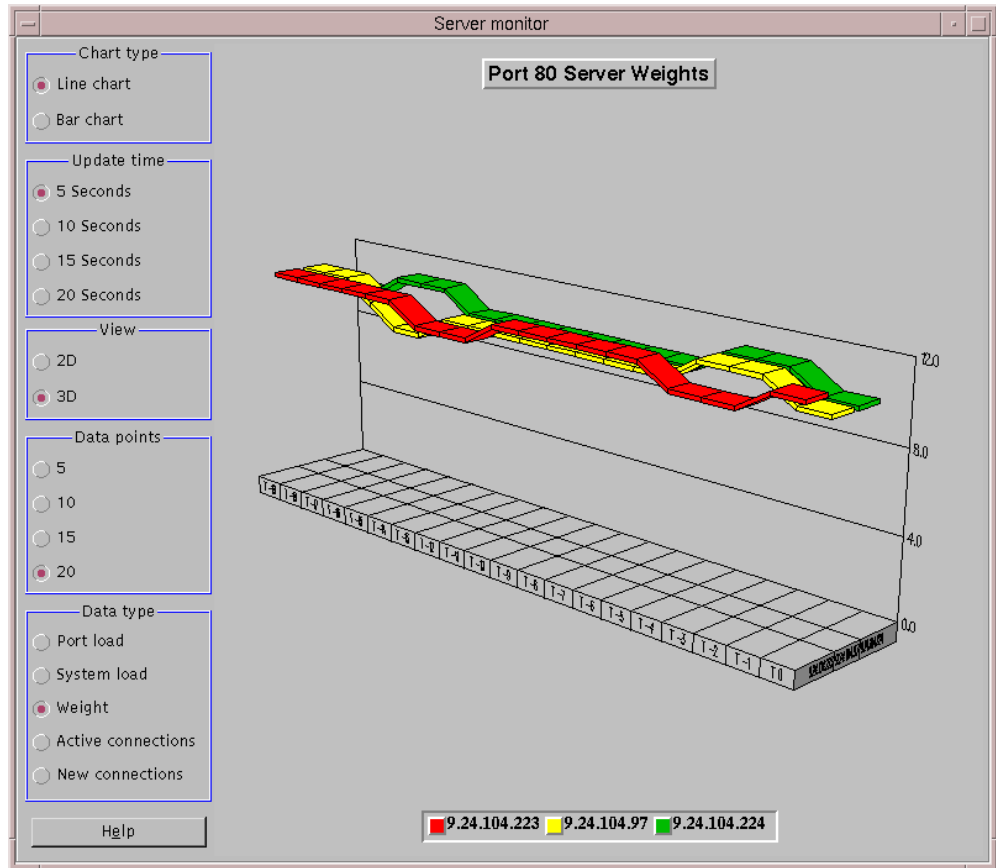


Figure 331. Servers' Weights 100 Clients

The above diagrams show the activity of the three Web servers during an interval of intense workload. They also demonstrate how the workload is distributed by the Dispatcher machine among all the available servers. The workload represented in the above diagrams was completely generated by dynamic requests, which were achieved by HTML pages invoking CGI-Bin programs. We verified that all the static HTML pages had already been cached by the caching proxy server, which was able to serve them from its own local cache. This means that workload would have been even higher, had we used a caching proxy server.

In the following three figures we show you the performance diagrams of this scenario. In order to show you how performances increase when using a clustered proxy server site, we report on the same figures also the diagrams related to the basic scenario, described in 6.1, "First Step: Regional and Local Access ISP Basic Scenario" on page 370:

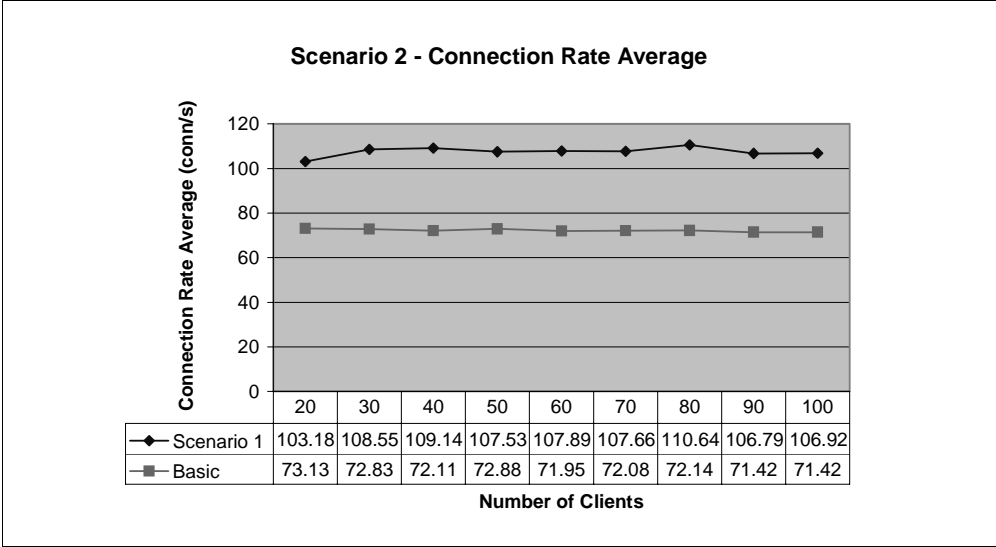


Figure 332. Connection Rate Average

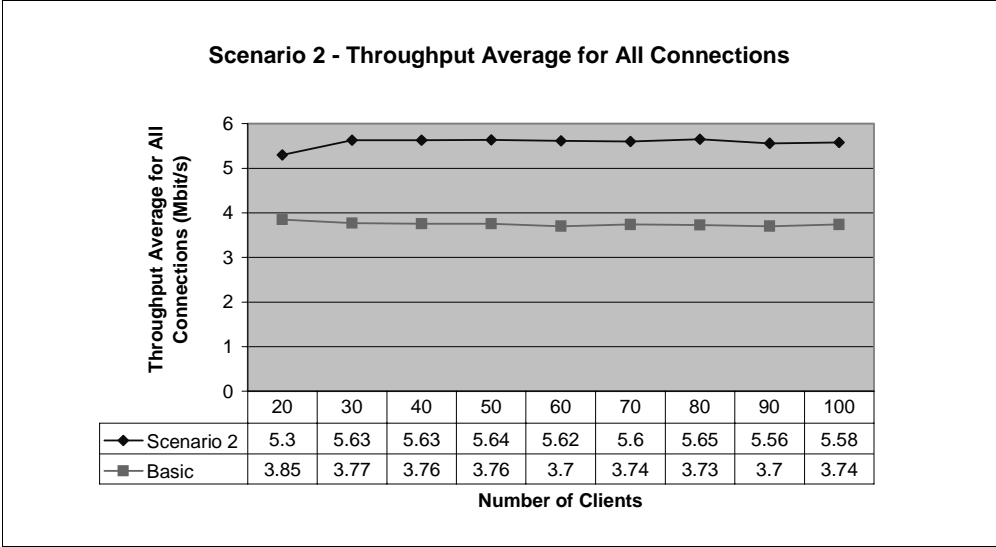


Figure 333. Throughput Average for All Connections

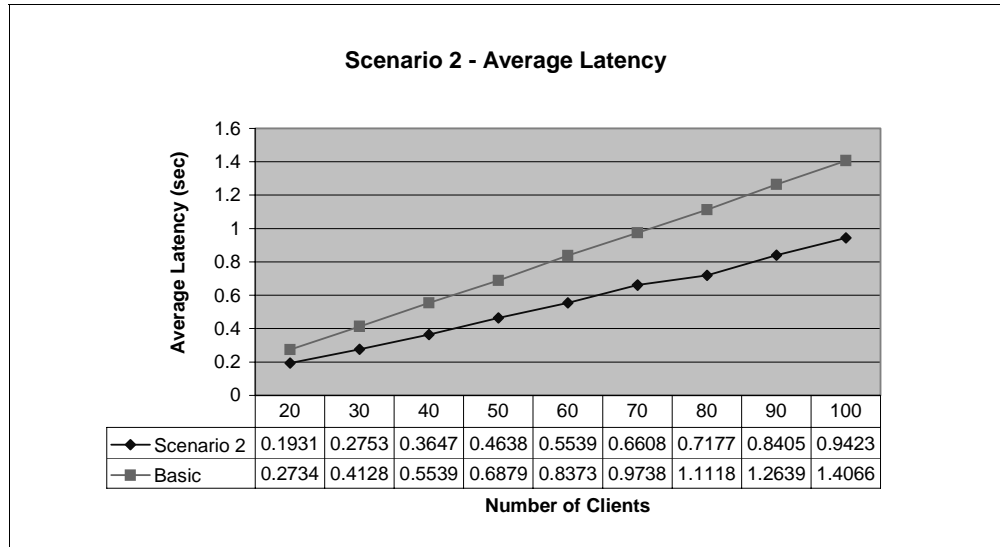


Figure 334. Average Latency

We remind you that the above results should not be considered the real performances of a Web environment where IBM WebSphere Performance Pack has been installed. The platform where we implemented our scenario was limited, as we explain in the note “Value of Performance Measurements” on page 417.

However, it is very interesting to see how in the regional and local access ISP architecture of our platform, even when 100 concurrent clients access the same Web site generating a light dynamic workload (25% of dynamic pages that invoke a CGI-Bin program, 75% of static HTML pages):

- The connection rate average, which is the average rate of creation and destruction of client/server connections, is about 107 connections per second.
- The throughput average for all connections, which is the average data transfer rate, is about 6 Mbps.
- The average latency, which is the average client wait for data to be returned, is less than 1 second.

These results are even more impressive if compared with the results found in 5.1, “First Step: Traditional Scenario” on page 350, 6.2, “Second Step: Regional and Local Access ISP Complete Scenario” on page 379 and Appendix B.5, “Scenario Example” on page 418.

However, they also demonstrate how limited our scenario environment was. IBM used the WebSphere Performance Pack technology to create a scalable and reliable system that efficiently handled unprecedented traffic volumes. On February 17th, 1998, the Official Web site of the Olympic Winter Games in Nagano made Internet history by logging a staggering 103,400 hits per minute, while still providing normal response time. The Internet site of the 1998 Nagano Olympic Winter Games is recognized by the Guinness Book of World Records. Instead, from Figure 332 on page 395, you can see that in our environment we reached a maximum connection rate average of about 6,500 connections per minute, which is 15 times less than the real power that IBM WebSphere Performance Pack would allow in a perfectly tuned environment. For this reason,

we remind you that our scenarios were only aimed at demonstrating how the various components of IBM WebSphere Performance Pack can be combined together to provide complex and powerful architectures. It is interesting to see how the performances of a certain Web site, although not optimized, increase when using IBM WebSphere Performance Pack. However, performance measurements in an environment as limited as ours do not have a real value.

Chapter 7. Large ISP Scenario

This chapter gets very specific on how to implement a large Internet service provider (ISP) environment. This scenario reflects the real situation of backbone ISPs, full service ISPs and international ISPs with other ISPs and large corporations as primary customers.

The architecture of the large ISP scenario we implemented is shown in the following diagram:

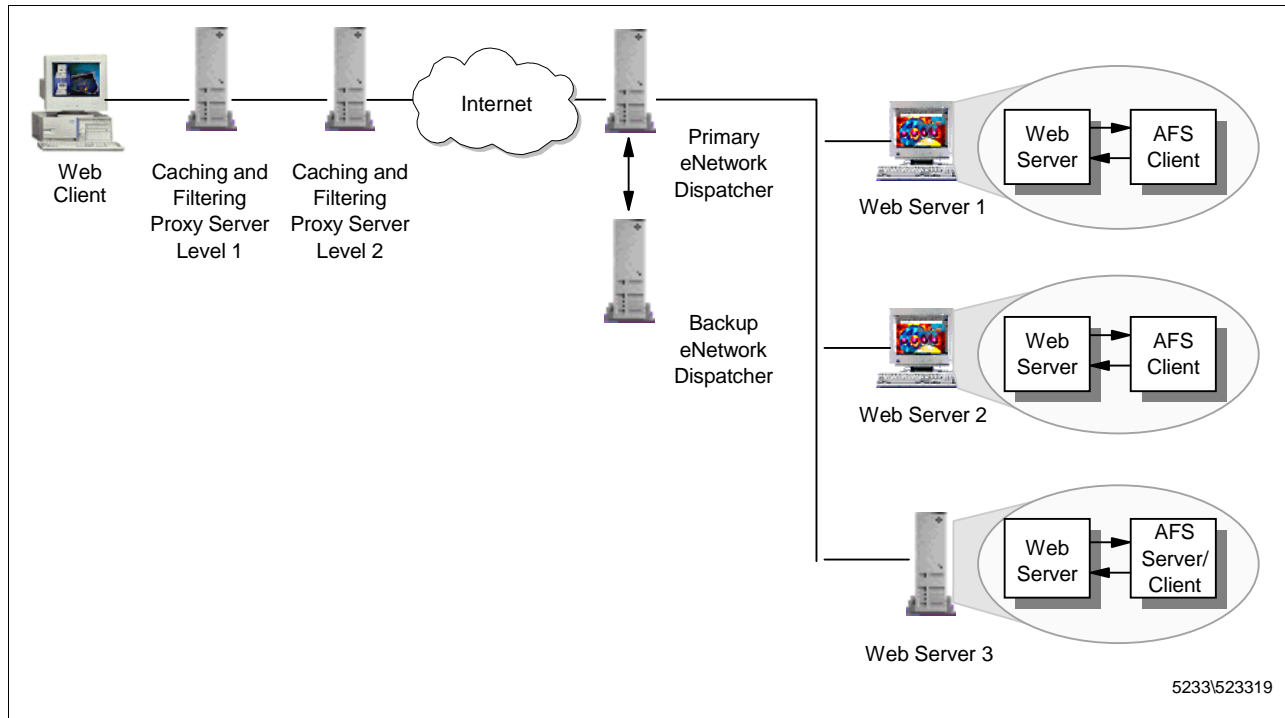


Figure 335. Large ISP Scenario Architecture

This architecture shows the real potentiality of the IBM WebSphere Performance Pack. With only one product you can build up a very complex network architecture, providing you all the most advanced functionalities requested nowadays to meet Internet/intranet needs, both from the server side and from client side.

As you can see, this scenario makes extensive use of all the components of IBM WebSphere Performance Pack:

1. Requests coming from client machines are intercepted by a caching and filtering proxy server, which serves the response directly to the client if the requested Web page has already been requested and cached, or it forwards the client's request to another caching proxy server, which is found at the second level in the proxy hierarchy. The second level proxy acts as the first one. It checks its own cache and if the response is not found there, forwards the request to the Web site.

In other words, this architecture implements proxy chaining, which is a very useful feature allowed by the Caching and Filtering component of IBM WebSphere Performance Pack.

The larger the number of the users, the higher the probability that the proxy server already has the document in its cache. A second level proxy usually serves more users, so it is likely that it has a lot of pages in its cache and the proxy at the lower level, which is closer to the client that originated the request, benefits from the cache of the higher level proxy.

Proxy chaining reduces the load on the higher level proxy (typically, the firewall proxy) and ultimately on the content server, since the lower level proxy may already have the document cached.

The caching and filtering proxy servers can also apply content filtering at the proxy server level, according to the PICS specifications.

For further details on proxy chaining, see 3.6.4, “Proxy Chaining” on page 120.

2. The Load Balancing component is used here to distribute the load between different Web servers.

Notice that this architecture implements *Dispatcher high availability*. A standby Dispatcher machine remains ready at all times to take over load balancing should the primary Dispatcher machine fail. The high availability configuration detects and recovers from network and server failures.

For further details on Dispatcher high availability, see 4.11, “Dispatcher High Availability” on page 313.

3. The File Sharing component allows the Web server machines to share the same Web content in a non-disruptive way. The AFS file system keeps data between all the Web servers consistent, while making it easy to update the data. It offers the opportunity of having the exact directory structure so that Web servers can have the same configuration files.

Moreover, combining the file sharing and load balancing functions in the same environment guarantees *Web server high availability*, meaning that if one of the Web servers should fail, the service would not be interrupted and the end user would not experience any problems. You can also easily implement *ISS high availability* if you decide to use the ISS function in your architecture (see 4.8.1, “ISS Configuration File” on page 295).

It is also possible to enhance the architecture shown in Figure 335 on page 399 adding, for example, caching proxy server load balancing as discussed in 6.2, “Second Step: Regional and Local Access ISP Complete Scenario” on page 379.

The following diagram offers a graphical representation of the application flow in a large ISP scenario:

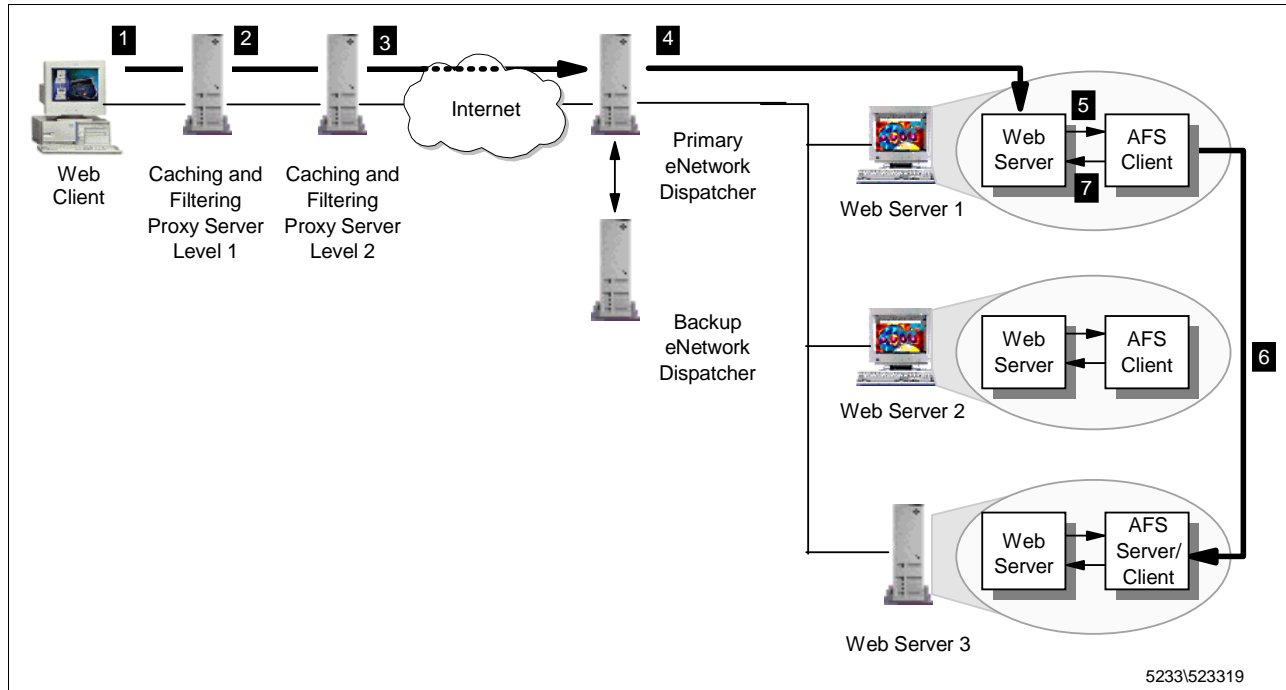


Figure 336. Flow Diagram of the Large ISP Scenario

The following list gives more details on the application flow graphically represented in the above diagram:

1. The Web client software issues a request using the cluster address of the Web site (for instance `http://9.24.104.105/afs/webstone/html/file2k.html`). Since the browser is configured to submit its requests to the IP address of the first level caching proxy server, the client's request is intercepted by the first level caching proxy server.
2. The first level caching proxy server serves the response back to the client if the requested Web page has already been cached. Otherwise, it forwards the request to the second level caching proxy server.
3. The second level caching proxy server serves the response back to the first level proxy server if the requested Web page has already been cached. Otherwise, it forwards the request to the Web site. Since the cluster address of the clustered Web site had been specified by the client, such a request is intercepted by the Dispatcher.
4. The Dispatcher selects the appropriate Web Server (in this example, Web Server 1) and forwards the request to it.
5. The `httpd` daemon running on Web Server 1 interprets the HTTP request and issues a local I/O call that is intercepted by the AFS Cache Manager to retrieve the data.
6. The Cache Manager retrieves the data from its local cache if the requested data is present and up-to-date, or requests it from the AFS server.
7. Upon retrieving data, the Cache Manager gives it back to the `httpd` daemon that is able to honor the client's request.

Notice that the response from the Web server flows directly to the second level proxy server that submitted the request, without passing through the Dispatcher of the clustered Web site (see 4.6.5, “How the Dispatcher Works: The Flow of the IP Packets” on page 273).

The hardware, software and network configuration of the large ISP scenario we implemented is shown in the following table:

Table 16. Large ISP Scenario - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	aixncf157	172.16.1.5	AIX 4.3.1	Web Client
IBM RS/6000 43P	rs600022	172.16.1.2	AIX 4.3.1	1 st level Caching Proxy Server
IBM RS/6000 43P	venus	172.16.1.3	AIX 4.3.1	2 nd level Caching Proxy Server
IBM RS/6000 43P	rs600012	I) 172.16.1.1 E) 9.24.104.124	AIX 4.3.1	IP Router
IBM RS/6000 43P	1) rs600023 2) clusterend	1) 9.24.104.128 2) 9.24.104.105	AIX 4.3.1	Primary Dispatcher
IBM RS/6000 43P	1) august 2) clusterend	1) 9.24.104.46 2) 9.24.104.105	AIX 4.3.1	Backup Dispatcher
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3.1	AFS Server and Client, Web Server
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	AFS Client, Web Server
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	AFS Client, Web Server

The role of the Web client was provided by WebStone 1.1, while the Web server functionality was provided by Lotus Domino Go Webserver 4.6.2.5. All the details on the configuration of the environment shown in the above table can be found in Appendix A, “Network Configuration of the Scenario Environments” on page 405. The configuration of WebStone can be found in Appendix B, “Scenario Basic Configuration” on page 411.

About the load balancing configuration of the clustered Web site, we defined a cluster of three servers on the TCP port 80, since we wanted the Dispatcher to load balance only HTTP requests.

Notice that this scenario is an enhancement from the scenario described in 6.1, “First Step: Regional and Local Access ISP Basic Scenario” on page 370. So you can safely repeat the same configuration steps, with two differences:

1. In the client-side, you should put two chained caching proxy servers to which clients’ requests are directed. To improve performances, both the caching proxy servers should have the caching function enabled.
2. On the server-side, you should add another Dispatcher machine with a backup role. It will start dispatching the requests only if the primary Dispatcher machine should fail.

Co-Location Option

Notice that the Load Balancing component can be installed on the same machine where one of the application servers reside. This feature is known as a co-location option and it is particularly useful, especially if you want your Web site to benefit from the high availability and scalability options of the Load Balancing component with a minimum investment.

These are the configuration steps in detail:

1. Set up all the machines (see Appendix A., “Network Configuration of the Scenario Environments” on page 405 and Appendix B., “Scenario Basic Configuration” on page 411).
2. Install and configure the two (or more) caching proxy servers (see 3.7, “Installation of the Caching and Filtering Component” on page 120 and 3.8, “Basic Configuration” on page 149).

We did not use the PICS filtering function in this architecture, but if you want to implement it in your environment, you can refer to 3.12, “PICS Scenario” on page 186.

3. Configure the proxy chaining function (see 3.9, “Proxy Chaining Scenario” on page 164).
4. Install and configure the two Dispatcher machines on the server-side (see 4.5, “Installation of the Load Balancing Component” on page 229 and 4.6.3, “Dispatcher Configuration” on page 248).

We did not use the ISS function in this architecture, but if you want to implement ISS in your environment you can refer to 4.8, “Load Balancing Scenario Using the Dispatcher and ISS Functions” on page 294.

5. Configure the Dispatcher high availability function (see 4.12, “Dispatcher High Availability Scenario” on page 314).
6. Configure the Web servers to be part of the clustered Web site and the caching proxy servers to be part of the clustered proxy server site (see 4.6.4, “TCP Servers Configuration” on page 267).
7. Install and configure the File Sharing component on the Web server machines (see 2.4, “Installation and Configuration of the File Sharing Component” on page 21).

The details of the Dispatcher configuration we implemented in the clustered Web site are shown in Figure 297 on page 358, Figure 298 on page 359, Figure 299 on page 360 and Figure 300 on page 361.

As all the scenarios that we have implemented and shown in Part 2, “IBM WebSphere Performance Pack Scenarios” on page 347, this one also is very significant from an architecture point of view. All these scenarios demonstrate how you can combine together all the components of IBM WebSphere Performance Pack to obtain reliable and scalable architectures.

Also in this case we used WebStone to generate the workload and verify that the architecture we built was running correctly. However, the way we ran WebStone would cause the two caching proxy servers to have exactly the same Web pages

in the caches. For this reason, performance results similar to those we have shown in the other scenarios would not make sense in this case.

Appendix A. Network Configuration of the Scenario Environments

This appendix contains details on the network configuration of the platform where we performed all the scenarios described in Part 2, "IBM WebSphere Performance Pack Scenarios" on page 347. We describe how we configured the network environment, in case you want to recreate our same experiences.

A.1 Computer Environment Configuration

We built up two token-ring LANs working at 16 Mbps, connected through a router. We call the LAN representing the client-side the *internal* LAN of our environment, and the other LAN, representing the server-side, the *external* LAN. The following tables summarize the hardware, software and network configuration of the environment where we performed our scenarios. However, consider that we did not use all the machines listed here in all the scenarios. Some scenarios involved a smaller number of machines. What we are showing here is the complete configuration. A clear explanation of which machines we effectively used in each single scenario is given in the chapter where that specific scenario is described.

This is the table that shows the configuration of the internal LAN. The internal LAN included, in its basic form, only the Web client machine on which WebStone ran. However, a more sophisticated IBM WebSphere Performance Pack scenario that we implemented involved a caching and filtering proxy server on the client-side, which is very useful to reduce the response time and the network bandwidth utilization. In even more sophisticated IBM WebSphere Performance Pack scenarios, we involved two caching and filtering proxy servers, load-balanced by the Load Balancing component or chained in a well-defined proxy hierarchy.

Table 17. Internal LAN - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	aixncf157	172.16.1.5	AIX 4.3.1	Web Client
IBM PC 365	1) wtr05219 2) clusternt	1) 172.16.1.4 2) 172.16.1.6	Windows NT Server 4.0	Load Balancing
IBM RS/6000 43P	rs600022	172.16.1.2	AIX 4.3.1	Caching and Filtering
IBM RS/6000 43P	venus	172.16.1.3	AIX 4.3.1	Caching and Filtering

The following table summarizes the configuration of the machine we used as the router between the internal and the external LANs. The router machine was equipped with two token-ring network adapters at 16 Mbps, and so it had two different IP addresses. 9.24.104.124 was the IP address in the external LAN, while the IP address in the internal LAN was 172.16.1.1.

Table 18. Router Machine - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	rs600012	I) 172.16.1.1 E) 9.24.104.124	AIX 4.3	IP Router

The configuration of the external LAN is shown in the following table. Notice that the server-side LAN involved in its basic configuration only a single Web server.

Then we increased the capacity of the server-side LAN by using three Web servers, which shared the same contents through the File Sharing component of IBM WebSphere Performance Pack and were balanced by a Load Balancing machine. We built an even more sophisticated scenario using two Load Balancing machines in high availability. The following table summarizes the configuration of the most complete scenario we implemented:

Table 19. External LAN - Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	1) rs600023 2) clusterend	1) 9.24.104.128 2) 9.24.104.105	AIX 4.3.1	Primary Dispatcher
IBM RS/6000 43P	rs600030	9.24.104.97	AIX 4.3	Web Server, AFS Server and Client
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	Web Server, AFS Client
IBM PC 365	computer1	9.24.104.224	Windows NT Server 4.0	Web Server, AFS Client
IBM RS/6000 43P	1) august 2) clusterend	1) 9.24.104.46 2) 9.24.104.105	AIX 4.3.1	Backup Dispatcher

Notice that the two Dispatcher machines listed in the above table ran the same operating system. This is a requirement when working with Dispatcher high availability.

On all the Windows NT machines listed so far, Service Pack 3 was applied.

The following table clarifies what products we used to implement the functions we have described:

Table 20. Software Products Used in the Scenarios

Web Client	WebStone 1.1, Netscape Navigator 4.6
Web Server	Lotus Domino Go Webserver 4.6.2.5
File Sharing	IBM WebSphere Performance Pack 1.0
Caching and Filtering	IBM WebSphere Performance Pack 1.0
Load Balancing	IBM WebSphere Performance Pack 1.0

A.2 IP Configuration

In our environment, to avoid performance reduction, we did not use a DNS server to provide host name resolution in the internal LAN. In this section we show the configuration we issued on the machines of our environment to ensure host name resolution without the use of any DNS servers.

A.2.1 External LAN

In the external LAN, all the machines kept using the already defined DNS server, which provided host name resolution for all the machines of the external life itself.

On each machine of the external LAN, we set as the default gateway as 9.24.104.124 (the router machine). This guaranteed the connection from the external LAN to the internal LAN.

On the Windows NT machines, we defined the default gateway from the TCP/IP Protocol Properties window.

On the AIX machines, we did that using smitty:

1. From a command line, enter the command `smitty`.
2. Select **Communications Applications and Services** and press Enter.
3. Select **TCP/IP** and press Enter.
4. Select **Further Configuration** and press Enter.
5. Select **Static Routes** and press Enter.
6. Select **Add a Static Route** and press Enter.
7. In the Add Static Route window:
 - Enter 172.16.1.0 as **DESTINATION** Address.
 - Enter 9.24.104.124 as **Default GATEWAY** Address.
 - Enter 255.255.255.0 as **Network MASK**.
 - Accept the default value 1 for **METRIC**.
8. Press Enter.
9. Exit smitty by pressing the F10 functional key.

After this configuration, to show the routing tables, issue the following command:

```
netstat -rn
```

We show you, as an example, the output we received on the AIX machine having hostname rs600023 and IP address 9.24.104.128, when we entered the above command:

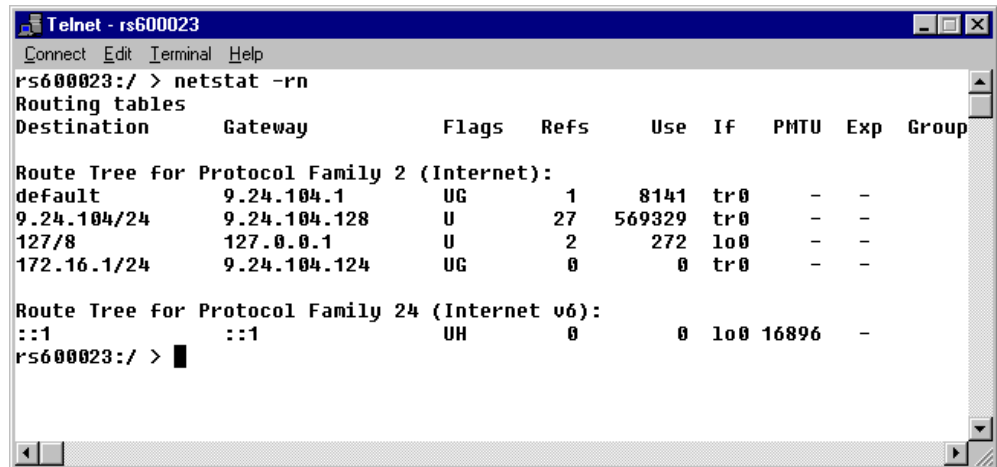


Figure 337. Display the Routing Tables on AIX Machine

A.2.2 Internal LAN

We made use of a local host name resolution file on the machines in the internal LAN. Such a file provides host name resolution by associating host names to IP addresses. If such a file is defined, a computer does not need to contact a DNS server to translate host names into IP addresses, and host name resolution is achieved locally. Notice that it is required that a host name resolution file on a given computer contains one entry for each machine the computer will need to contact through the host name.

The fully qualified host name resolution file name is the following:

- /etc/hosts on AIX
- C:\WINNT\system32\drivers\etc\HOSTS on Windows NT

The syntax that both Windows NT and AIX systems recognize for this file is:

```
host_name IP_address # comments
```

We added the following lines to the host name resolution files:

```
#Resolution of inner-lan machine
172.16.1.1          rs600012
172.16.1.2          rs600022
172.16.1.3          venus
172.16.1.4          wtr05219
172.16.1.5          aixncf157
172.16.1.6          clusternt

#Resolution of outer-lan machine
9.24.104.97         rs600030.itso.ral.ibm.com
9.24.104.223        wtr05195.itso.ral.ibm.com
9.24.104.105        clusterend.itso.ral.ibm.com
9.24.104.46         august.itso.ral.ibm.com
9.24.104.224        computer1.itso.ral.ibm.com
9.24.104.128        rs600023.itso.ral.ibm.com
```

Figure 338. Host Name Resolution File for Computers in the Internal LAN

Notice that we edited and saved the above file only in the machines of the internal LAN. For this reason the IP address 9.24.104.124 corresponding to the external token-ring network interface on the router machine rs600012 does not appear in the host name resolution file.

To disable interactions with DNS servers, we issued the following command on all the machines in the internal line:

```
mv /etc/resolv.conf /etc/resolv.conf.back
```

Then, in the TCP/IP configuration of each machine, we defined a static route with destination address 9.24.104.0, network mask 255.255.255.0 and default gateway 172.16.1.1, which was the IP address assigned to the internal token-ring network interface on the IP router machine. This was done in a way similar to what is shown in A.2.1, “External LAN” on page 406.

A.2.3 IP Router

The IP router machine was a RISC/6000 having two token-ring network adapters. The network interface in the internal LAN had been given the IP address 172.16.1.1, while the interface on the external LAN had IP address 9.24.104.124.

Remember to set `ipforwarding=1` to enable that machine to route packets between the two LANs. We added in the `/etc/rc.tcpip` file the following line:

```
no -o ipforwarding=1
```

On this machine, we did not disable DNS host name resolution for machines in the external LAN, and we did not edit the `/etc/hosts` file. It kept using the DNS server of the external LAN, which provided host name resolution for all the machines of the external LAN. This machine was unable to resolve host names in the internal LAN, but this was not a problem in our environment, because:

- Machines of the internal LAN were configured to perform host name resolution locally, without using any DNS servers.
- Machines in the external LAN would contact the machines of the internal LAN only through the IP address, not through the host name.

A.3 TCP/IP and Adapter Tuning

We also made some modifications to get better performances. We made these modification on the WebStone machine and on the first AFS machine, which were two RISC/6000 machines installed with AIX and provided with a token-ring network interface.

A.3.1 Maximum Transmission Unit

We changed the Maximum Transmission Unit (MTU) parameter. A larger MTU will result in higher connections per second. We entered:

```
smitty netinterface
```

On the Network Interfaces smitty screen, we selected **Network Interface Drivers** and pressed Enter. The Available Network Interfaces screen was displayed and we selected **tr0**. Then we set the parameter `Maximum IP PACKET SIZE for THIS DEVICE` to 2000.

A.3.2 The `tcp_sendspace` and `tcp_recvspace` Parameters

We changed the default values of `tcp_sendspace` and `tcp_recvspace`, which were set to 16KB. The new value we set was 64KB. This we did by issuing the following two commands:

```
no -o tcp_sendspace=64000
no -o tcp_recvspace=64000
```

Then, to make sure that these settings would be maintained after each reboot, we added the above commands in the file `etc/rc.tcpip`.

This operation also increase the number of connections per second.

A.3.3 The MBUFS Parameter

Last, we increased the value of the `MBUFS` parameter. The default was 0 KB, and we changed it to 32 KB.

To do this, we entered `smitty`, then we selected **System Environments and Change / Show Characteristics of Operating System**. In the page we got, we could set the value of the parameter `Maximum Kbytes of real memory allowed for MBUFS` to 64.

A.4 Increasing the Performances on the Web Client Machine

The settings we have shown in A.3, “TCP/IP and Adapter Tuning” on page 409 were performed on both the client machine, running WebStone, and the first AFS machine, which was both an AFS server and client and where the `httpd` daemon was running.

In this section we show you what we did to improve further the performances of the client machine. As explained in Appendix B, “Scenario Basic Configuration” on page 411, WebStone is a tool that generates a high number of processes and simulates multiple concurrent clients that submit requests to a Web site. So it is a good idea to increase on the Web client machine the maximum number of processes allowed per user, which by default is 40, and the paging space.

A.4.1 Number of Processes per User

To increase the number of processes allowed per user, we selected **System Environments and Change / Show Characteristics of Operating System**. In the page we got, we set the value of the parameter `Maximum number of PROCESSES allowed per user` to 100.

A.4.2 Paging Space

We also increased the paging space on the client machine where WebStone would run.

To do this, we followed the same procedure indicated in 3.7.1, “Installation on AIX” on page 120. The paging space we set was 200MB.

Appendix B. Scenario Basic Configuration

This appendix gives more details on the scenarios we built using IBM WebSphere Performance Pack, that are described in Part 2, “IBM WebSphere Performance Pack Scenarios” on page 347. The details that you will find in this appendix are not strictly related to IBM WebSphere Performance Pack, but can be very helpful if you want to recreate the same experiences that we did. This appendix also contains some useful recommendations to increase the performances of a Web server machine.

In each scenario, the server-side platform is completely based on IBM WebSphere Performance Pack and its components. For this reason, we gave a complete description on how to configure the server-side platform in the chapters of Part 2, “IBM WebSphere Performance Pack Scenarios” on page 347.

The client-side, in a real-life scenario, is composed of multiple Web clients, and does not require particular settings. In order to generate a consistent Web load and study the performances of the Web sites we built, we used particular tools and system configurations. In this appendix we illustrate to you the basic configuration on the client-side platform we implemented, and we give you all the details of the tools we used to generate a consistent Web load and study the performances of the Web sites we built in the various scenarios.

B.1 Installation of the Client Platform

The client platform of our scenarios was composed of one single machine, where WebStone 1.1 was running. WebStone is a benchmark for Web servers provided by Silicon Graphics. It is able to simulate the presence of a number of clients that simultaneously sends requests to a Web site.

When this book went to print WebStone Version 2.5 became available. We did not have the opportunity to use the last version. However, our main purpose was to generate a consistent Web load and WebStone Version 1.1 matched our needs. The most up-to-date version of WebStone can be downloaded for free from <http://www.mindcraft.com>.

This section shows how we installed the client machine running WebStone 1.1. We also give hints and tips for people who intend to recreate the same experiments.

The scenarios we built were based on a token-ring network at 16 Mbps. We installed WebStone on a RISC/6000 Model 43P, with 196 MB of RAM, 3.2 GB of hard disk and 133 MHz of CPU. This machine was installed with AIX 4.3.1. This machine was equipped with a token-ring adapter that worked at 16 Mbps. The fully qualified host name of this machine was aixncf157.itso.ral.ibm.com and the IP address was 172.16.1.5.

The installation of WebStone 1.1 on AIX is very simple. We downloaded the installation file WebStone-1.1.tar and placed it in the directory /home/guest of the client machine aixncf157. To untar it, we issued the command:

```
tar -xvf WebStone-1.1.tar
```

B.2 Web Servers Configuration to Support WebStone's Workload

On the client machine we found three interesting directories that had been generated during the installation of WebStone:

- /home/guest/WebStone-1.1/html, containing all the HTML workload files
- /home/guest/WebStone-1.1/bin, containing all the WebStone's binary files
- /home/guest/WebStone-1.1/conf, containing all the configuration files

Even though in the scenarios described in Part 2, "IBM WebSphere Performance Pack Scenarios" on page 347, we often used three Web servers, the Web contents were actually stored only in one of them, which was also an AFS server. The other two Web servers ran the httpd daemon, but did not store the contents. In fact they were configured as Web clients and could share the contents stored in the AFS server machine. This was possible because an AFS client sees shared files as local files, and a process, such as the httpd daemon, running on an AFS client machine can access the shared files as if they were local.

The workload HTML and CGI-Bin files generated in the client machine after the installation of WebStone must be copied in the Web server machine that is also AFS server. It is not necessary that the other two Web servers host a physical copy of each HTML file, since they can simply share the contents hosted on the AFS server.

On the AFS server machine of our platform, which was also an AFS client (see 2.4.1, "Installation of the First AFS Machine on AIX" on page 22), we copied the workload HTML and CGI-Bin files. We created a subdirectory webstone under the path /afs/itso.ral.ibm.com/afsuser, which was the home directory of the AFS user afsuser (see 2.5.5, "User's Volume Mount Point" on page 88). Then under the directory webstone we created two subdirectories, html and cgi-bin. We copied all the workload HTML files under /afs/itso.ral.ibm.com/afsuser/webstone/html. We needed only one CGI-Bin file, named cgi-send.cgi.AIX, which is specific for the AIX platform. (The others are for Web servers running on other operating systems.) So we copied that file under /afs/itso.ral.ibm.com/afsuser/webstone/cgi-bin.

Lotus Domino Go Webserver 4.6.2.5 was the product we installed on the AFS machines to provide them with the Web server functionality. To hide the effective path where the WebStone files were located, we added the following entries in the Lotus Domino Go Webserver configuration files of all the Web servers:

```
Pass /afs/*           /afs/itso.ral.ibm.com/afsuser/*
Pass /webstone/*     /afs/itso.ral.ibm.com/afsuser/webstone/html/*
Pass /webstone/cgi-bin/* /afs/itso.ral.ibm.com/afsuser/webstone/cgi-bin/*
```

The Lotus Domino Go Webserver configuration file is by default /etc/httpd.conf on AIX and C:\WINNT\httpd.cnf on Windows NT.

B.3 Configuration of the Client Platform

In this section we describe step by step the configuration of the client machine where we installed WebStone.

B.3.1 Creation of the File `/.rhosts`

When it runs, WebStone needs to make a remote shell log in on the same machine where it has been installed. To accomplish this task, an ASCII file named `.rhosts` must be present in the home directory of the user that runs WebStone, and that file must contain the host name of the WebStone machine. On our platform, since we ran WebStone as root, we created the file `.rhosts` under the root directory `/`, by entering:

```
vi /.rhosts
```

Then we inserted the following line in the `.rhosts` file:

```
aixncf157
```

We then saved the file. Notice that `aixncf157` was the host name of the machine where we installed WebStone.

B.3.2 Disabling DNS

The WebStone machine was part of the internal LAN in our environment. For this reason, from a DNS perspective, we applied to it the same configuration we have seen for the machines of the internal LAN (see A.2.2, “Internal LAN” on page 408).

B.3.3 WebStone Configuration Files

Before you run WebStone, you need to set up two files on the WebStone machine:

- The first one, named `filelist`, defines the list of workload files that WebStone will request to the Web site.
- The second one, named `testbed`, defines configuration directives.

Both the files are located under the directory `/home/guest/WebStone-1.1/conf/`. Actually, in that directory, there are multiple `filelist` files, each one with a different extension, depending on the kind of load you want to generate: static, light dynamic, medium dynamic and heavy dynamic. Depending on what `filelist` file you select, the percentages of CGI-Bin and static HTML files that will be requested to the Web site will vary. We decided to configure WebStone to generate a dynamic light workload, since we considered this configuration very close to a real-life situation. To do this, we selected the file `filelist.dynamic-light`, whose default contents are shown next:

```
8
40 2
/file2k.html
/cgi-bin/cgi-send.cgi?send=3072
25 2
/file1k.html
/file5k.html
15 2
/file4k.html
/file6k.html
5 1
/cgi-bin/cgi-send.cgi?send=7168
4 4
/file8k.html
/file9k.html
/file10k.html
/file11k.html
4 5
/file12k.html
/file14k.html
/file15k.html
/file17k.html
/file18k.html
6 1
/file33k.html
1 1
/file200k.html
```

Figure 339. /home/guest/WebStone-1.1/conf/filelist/filelist.dynamic-light

The first number that appears in the file, 8, means that eight pages follow. Then you can see a couple of numbers (40 2), which mean that the first page will be requested 40% of the times and will consist of the static request /file2k.html and the dynamic request /cgi-bin/cgi-send.cgi?send=3072.

The name convention is as follows:

- File names such as filexk.html indicate HTML files of x KB.
- cgi-send.cgi is an executable that accepts a size parameter. This parameter must be set to a number y, and then the executable generates a random file of y bytes to return to the requester.

The rest of the entries in the above file can be explained in a very similar way.

In order to permit WebStone to find the correct path to retrieve the files, the above file /home/guest/WebStone-1.1/conf/filelist/filelist.dynamic-light must be customized according to the environment platform you set up. The following figure shows the file /home/guest/WebStone-1.1/conf/filelist/filelist.dynamic-light after the modifications we made:

```
8
40 2
/afs/webstone/html/file2k.html
/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072
25 2
/afs/webstone/html/file1k.html
/afs/webstone/html/file5k.html
15 2
/afs/webstone/html/file4k.html
/afs/webstone/html/file6k.html
5 1
/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=7168
4 4
/afs/webstone/html/file8k.html
/afs/webstone/html/file9k.html
/afs/webstone/html/file10k.html
/afs/webstone/html/file11k.html
4 5
/afs/webstone/html/file12k.html
/afs/webstone/html/file14k.html
/afs/webstone/html/file15k.html
/afs/webstone/html/file17k.html
/afs/webstone/html/file18k.html
6 1
/afs/webstone/html/file33k.html
1 1
/afs/webstone/html/file200k.html
```

Figure 340. Modified File `/home/guest/WebStone-1.1/conf/filelist/filelist.dynamic-light`

As you can see, we only modified the path where WebStone would find the files in the Web server machines. We also specified that the executable would be `/cgi-send.cgi.AIX`, which is the one specifically used on the AIX platform.

Notice that in the above file the files are listed through their partial URLs, and the host name of the Web server is not specified. This is possible because WebStone retrieves the host name of the Web server from the second important configuration file, `home/guest/WebStone-1.1/conf/testbed`, which is shown in the next figure:

```

ITERATIONS="1"
MINCLIENTS="20"
MAXCLIENTS="100"
CLIENTINCR="10"
TIMEPERRUN="2"
SERVER="rs600030"
PORTNO=80
WEBSTONEROOT="/home/guest/WebStone-1.1"
CLIENTS="aixncf157"
CLIENTACCOUNT=root
CLIENTPASSWORD=pistoia
TMPDIR=/tmp
RCP="rcp"
RSH="rsh"

```

Figure 341. Configuration File `home/guest/WebStone-1.1/conf/testbed`

With this configuration file, WebStone starts the requests to the server with 20 clients (`MINCLIENTS`), and for 2 minutes (`TIMEPERRUN`) it retrieves documents from the server. After these 2 minutes, it increments in 10 the number of clients (`CLIENTINCR`), and so on, until it arrives to 100 clients (`MAXCLIENTS`). In our configuration, the value of the entry `SERVER` was the host name of the Web server where the files to retrieve were located, while the value of `CLIENTS` was the host name of the machine where WebStone was installed. The two parameters `CLIENTACCOUNT` and `CLIENTPASSWORD` are respectively the user ID and password for the user that runs WebStone. We ran Webstone as the user root.

Notice that the value of the entry `SERVER` strictly depends on which IBM WebSphere Performance Pack architecture you are implementing:

- It must be set to a Web server host name if the client interacts directly with a Web server.
- If the requests go to a Dispatcher first, and then the Dispatcher selects the best available Web server, the value of `SERVER` must be set to the Dispatcher cluster address.
- If the requests go to a proxy, `SERVER` must indicate the host name of the proxy server, but in this case the entries in the filelist must include fully qualified URLs, such as:

```
http://rs600030/afs/webstone/html/file2k.html
```

Not simply:

```
/afs/webstone/html/file2k.html
```

We set the parameters in the testbed file according to the standard rules that are recommended at the WebStone Web site <http://www.mindcraft.com/webstone>.

B.3.4 Launching WebStone

Before launching WebStone, it is necessary to create the directory `/usr/local/bin`, where WebStone will store an executable file, named `webclient`, used to set up the client processes:

```
mkdir /usr/local/bin
```

Then WebStone can be launched through the following command:

```
/home/guest/WebStone-1.1/webstone
```

B.4 Information Provided by WebStone

We used WebStone with the main purpose of generating multiple requests simultaneously, in order to simulate a real-life situation. In fact WebStone creates a load on a Web server by simulating the activity of multiple Web clients, which can be thought of as users, Web browsers or other software that retrieves files from a Web server. This simulation is carried out using multiple clients running simultaneously on one or more computers.

However, WebStone does something more, since it gives back a list of values that can be interpreted to measure the performances of the Web site.

Value of Performance Measurements

When we built our scenarios, our main purpose was *not* to measure the performances of the Web sites we had created. Our scenarios were aimed to demonstrate how the various components of IBM WebSphere Performance Pack can be combined together to provide complex and powerful architectures. We implemented such architectures and we tried them simulating a workload with WebStone.

Performance measurements in an environment as limited as ours do not have a real value. We could have used more powerful machines and a larger number of clients, tuned our machines to get a better utilization of CPU and a faster disk access, used a faster network, etc. However, keeping in mind these limitations, we think it is interesting to see how the performances of a certain Web site, although not optimized, increase when using IBM WebSphere Performance Pack.

The following are the most significant parameters that WebStone lets you use to measure the performances of a Web site:

- Connection rate average

This parameter indicates the average rate of creation and destruction of client/server connections.

- Throughput average for all connections

This parameter indicates the average data transfer rate in Mbps.

- Little's load factor

This parameter is based on the theory of queues. It reflects how much time is spent on request processing. The Little's Load Factor is also an indirect indicator of the average number of connections that the Web site has open at a particular time. This number should stay very close to the number of clients, or else some clients are being denied access to the server at any given time.

- Average latency

This parameter indicates the average client wait for data to be returned.

- Errors rate

This parameter indicates the number of errors per second.

B.5 Scenario Example

This section shows an example of how to build a fully working scenario using WebStone to generate the load.

The scenario that we are showing here has not been reported in Part 2, “IBM WebSphere Performance Pack Scenarios” on page 347. Our purpose here is to show how to implement a fully working scenario using WebStone. However, this scenario is significant also for its relationship with IBM WebSphere Performance Pack, in that it makes use of the Caching and Filtering component, so you might be interested in understanding its architecture better.

A client machine, running WebStone, is configured to submit its request to a caching and filtering proxy server. These two machines are located in the same LAN segment. The flow of this scenario is described in the following list:

1. The Web client machine submits a request for a Web page located in a Web server. This request is submitted to a caching and filtering proxy server.
2. If the document that has been requested is already located in the cache of the proxy server, then the caching and filtering proxy server serves it from its local cache. Otherwise, the request is forwarded to the Web server, located on the Internet.

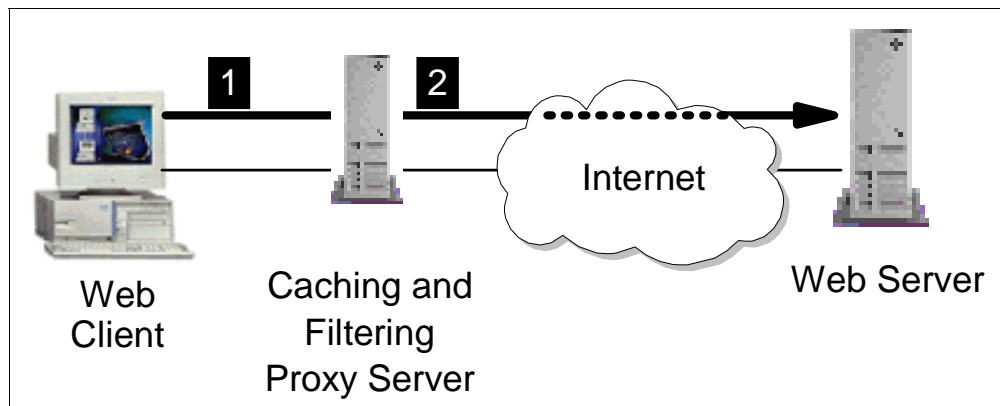


Figure 342. Graphical Representation of the Scenario Flow

Notice that the Web server responds to the proxy server, which caches the page, if this is cacheable, and then serves the response back to the client.

In our implementation of the above scenario, the two networks were separated by a router. The following table provides all the details about the network, hardware and software configuration:

Table 21. Hardware, Software and Network Configuration

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	aixncf157	172.16.1.5	AIX 4.3.1	Web Client
IBM RS/6000 43P	rs600022	172.16.1.2	AIX 4.3.1	Caching and Filtering

Workstation	Host Name	IP Address	Operating System	Service
IBM RS/6000 43P	rs600012	I) 172.16.1.1 E) 9.24.104.124	AIX 4.3	IP Router
IBM PC 365	wtr05195	9.24.104.223	Windows NT Server 4.0	Web Server

The caching proxy server functionality was provided by the Caching and Filtering component of IBM WebSphere Performance Pack 1.0. For further details on how we configured each single machine, refer to Appendix A, "Network Configuration of the Scenario Environments" on page 405.

The following figure shows the filelist we used in this scenario. As we have already explained in B.3.3, "WebStone Configuration Files" on page 413, it is necessary to specify the fully qualified URL when the requests must pass through a proxy server:

```

8
40 2
http://wtr05195/afs/webstone/html/file2k.html
http://wtr05195/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=3072
25 2
http://wtr05195/afs/webstone/html/file1k.html
http://wtr05195/afs/webstone/html/file5k.html
15 2
http://wtr05195/afs/webstone/html/file4k.html
http://wtr05195/afs/webstone/html/file6k.html
5 1
http://wtr05195/afs/webstone/cgi-bin/cgi-send.cgi.AIX?send=7168
4 4
http://wtr05195/afs/webstone/html/file8k.html
http://wtr05195/afs/webstone/html/file9k.html
http://wtr05195/afs/webstone/html/file10k.html
http://wtr05195/afs/webstone/html/file11k.html
4 5
http://wtr05195/afs/webstone/html/file12k.html
http://wtr05195/afs/webstone/html/file14k.html
http://wtr05195/afs/webstone/html/file15k.html
http://wtr05195/afs/webstone/html/file17k.html
http://wtr05195/afs/webstone/html/file18k.html
6 1
http://wtr05195/afs/webstone/html/file33k.html
1 1
http://wtr05195/afs/webstone/html/file200k.html

```

Figure 343. WebStone Filelist

The testbed configuration file is shown below:

```
ITERATIONS="1 "  
MINCLIENTS="20 "  
MAXCLIENTS="100 "  
CLIENTINCR="10 "  
TIMEPERRUN="2 "  
SERVER="rs600022 "  
PORTNO=80  
WEBSTONEROOT="/home/guest/WebStone-1.1 "  
CLIENTS="aixncf157 "  
CLIENTACCOUNT=root  
CLIENTPASSWORD=pistoia  
TMPDIR=/tmp  
RCP="rcp"  
RSH="rsh"
```

Figure 344. testbed Configuration File

Notice the following line:

```
SERVER="rs600022"
```

This shows that the `SERVER` variable must be set to the host name of the proxy server, if you want that the client's requests passed through the proxy.

After setting the above files, we launched WebStone through the command:

```
/home/guest/WebStone-1.1/webstone
```

After WebStone stopped, we saw that all WebStone's reports were reported under the directory `/home/guest/WebStone-1.1/bin/runs`. We found in that directory a large amount of redundant information. In order to see only the data we mentioned in B.4, "Information Provided by WebStone" on page 417, we used two WebStone's utilities, located in `/home/guest/WebStone-1.1/bin`. These utilities are named `wscollect` and `tabs2html`. We entered the two following commands:

```
/home/guest/WebStone-1.1/bin/wscollect runs > file.run  
/home/guest/WebStone-1.1/bin/tabs2html file.run > file.html
```

The first command reads all WebStone output data generated in the directory `/home/guest/WebStone-1.1/bin/runs`, extracts the most significant information and puts it in a file named `file.run`. The second command reads `file.run` and rewrites it in HTML format, generating a file named `file.html`.

This is an example of an HTML report:

Timestamp	Total number of clients	Connection rate average (conn/s)	Little's Load Factor	Average Latency (sec)	Error Rate (err/s)	Thruput average for all connections (Mbit/s)
980926_1418	20	44.17	20.02	0.4533	0.0000	2.39
980926_1421	30	46.92	30.02	0.6398	0.0000	2.54
980926_1424	40	39.66	40.10	1.0112	0.0000	2.03
980926_1427	50	38.52	50.21	1.3033	0.0000	2.08
980926_1431	60	39.00	60.41	1.5490	0.0000	2.07
980926_1435	70	38.32	70.37	1.8366	0.0000	2.02
980926_1440	80	40.92	80.40	1.9647	0.0000	2.20
980926_1445	90	39.83	91.07	2.2868	0.0000	2.19
980926_1450	100	38.21	101.09	2.6460	0.0000	1.91

Figure 345. HTML Sample WebStone Report

Among all the parameters that WebStone analyzes, we considered the throughput average for all connections, average latency and connection rate average as the most significant information. For this reason, after every scenario, we extracted the values reported for these three parameters and built three diagrams, in order to have a graphical understanding of the performances:

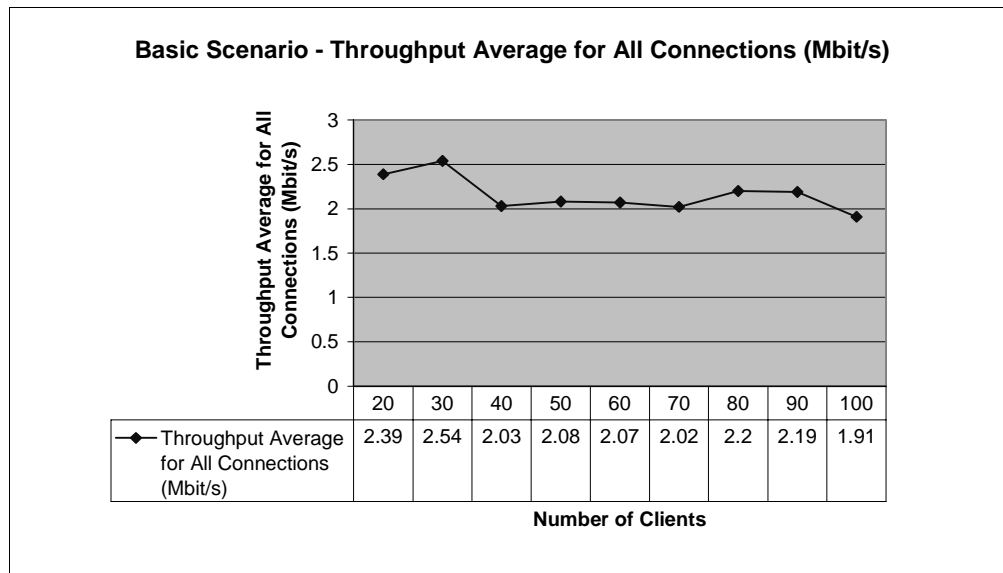


Figure 346. Basic Scenario - Throughput Average for All Connections

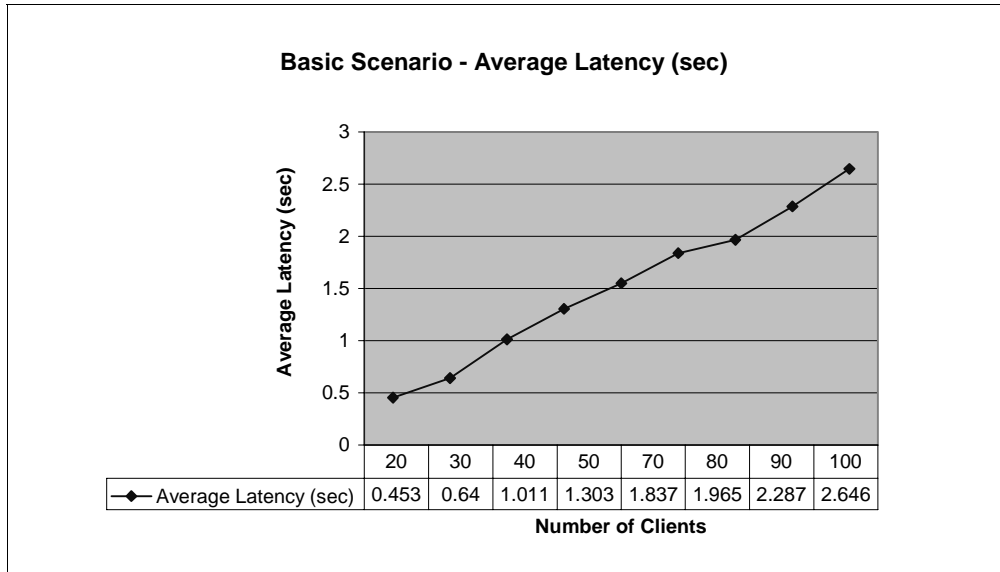


Figure 347. Basic Scenario - Average Latency

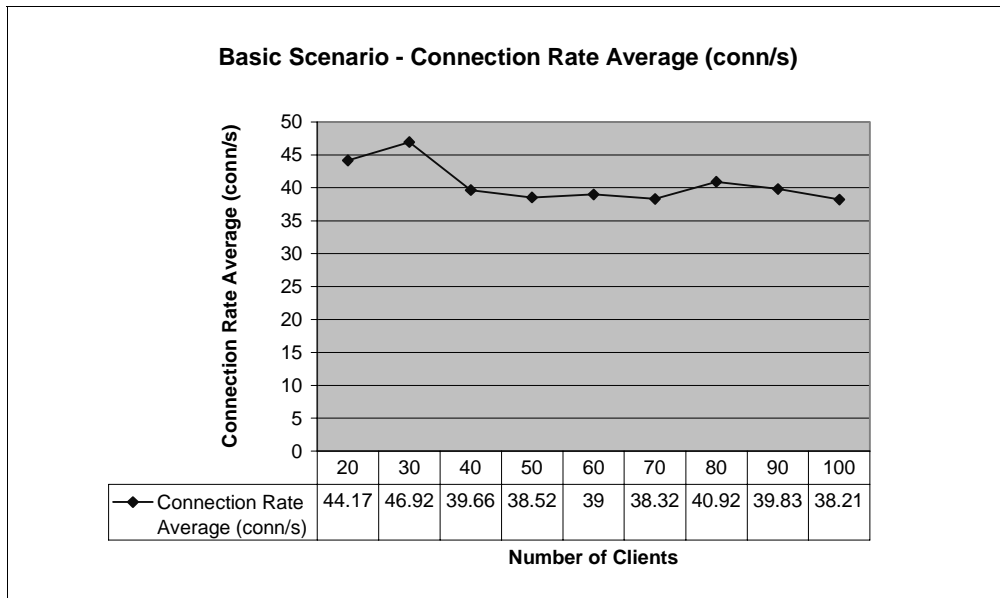


Figure 348. Basic Scenario - Connection Rate Average

As we expected, the performances were very good. In fact, if you look again at the filelist shown in Figure 343 on page 419, you can see that WebStone always requests the same files. In this case the requests are forwarded to the caching proxy server. The proxy server, in turn, forwards the requests to the Web server if the requested pages are not cached. This means that in a short time the proxy will find in its cache all the static files, which are cacheable. We can understand, in this way, how a caching proxy server can increase the performances. However, the caching proxy server cannot cache the dynamic pages; for these it will always forward an explicit request to the Web server.

Appendix C. Special Notices

This publication is intended to help customers, system engineers and I/T architects to plan for, install, configure, use, tune and troubleshoot each single component of IBM WebSphere Performance Pack Version 1.0. The information in this publication is not intended as the specification of any programming interfaces that are provided by IBM WebSphere Performance Pack. See the PUBLICATIONS section of the IBM Programming Announcement for IBM WebSphere Performance Pack for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM ("vendor") products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating

environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

IBM ®	DB2
DB2 Universal Database	RS/6000
eNetwork	ThinkPad
OS/2	OS/390

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names may be trademarks or service marks of others.

Appendix D. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

D.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 427.

- *Internet Security in the Network Computing Framework*, SG24-5220
- *Network Computing Framework Component Guide*, SG24-2119
- *Load-Balancing Internet Servers*, SG24-4993
- *RS/6000 Performance Tools in Focus*, SG24-4989
- *Understanding IBM RS/6000 Performance and Sizing*, SG24-4810
- *TCP/IP Tutorial and Technical Overview*, GG24-3376

D.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
Lotus Redbooks Collection	SBOF-6899	SK2T-8039
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043
Application Development Redbooks Collection	SBOF-7290	SK2T-8037

D.3 Other Publications

These publications are also relevant as further information sources:

- *eNetwork Dispatcher for Solaris, Windows NT, and AIX User's Guide*, GC31-8496
- *IBM Web Traffic Express for Multiplatforms User's Guide*, GC31-8645

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at <http://www.redbooks.ibm.com/>.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Redbooks Web Site on the World Wide Web**

<http://w3.itso.ibm.com/>

- **PUBORDER** – to order hardcopies in the United States

- **Tools Disks**

To get LIST3820s of redbooks, type one of the following commands:

```
TOOLCAT REDPRINT
TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
```

To get BookManager BOOKs of redbooks, type the following command:

```
TOOLCAT REDBOOKS
```

To get lists of redbooks, type the following command:

```
TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
```

To register for information on workshops, residencies, and redbooks, type the following command:

```
TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
```

- **REDBOOKS Category on INEWS**

- **Online** – send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (<http://www.redbooks.ibm.com/redpieces.html>). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** – send orders to:

In United States
In Canada
Outside North America

IBMMAIL
usib6fpl at ibmmail
caibmbkz at ibmmail
dkibmbsh at ibmmail

Internet
usib6fpl@ibmmail.com
lmannix@vnet.ibm.com
bookshop@dk.ibm.com

- **Telephone Orders**

United States (toll free)
Canada (toll free)

1-800-879-2755
1-800-IBM-4YOU

Outside North America
(+45) 4810-1320 - Danish
(+45) 4810-1420 - Dutch
(+45) 4810-1540 - English
(+45) 4810-1670 - Finnish
(+45) 4810-1220 - French

(long distance charges apply)
(+45) 4810-1020 - German
(+45) 4810-1620 - Italian
(+45) 4810-1270 - Norwegian
(+45) 4810-1120 - Spanish
(+45) 4810-1170 - Swedish

- **Mail Orders** – send orders to:

IBM Publications
Publications Customer Support
P.O. Box 29570
Raleigh, NC 27626-0570
USA

IBM Publications
144-4th Avenue, S.W.
Calgary, Alberta T2P 3N5
Canada

IBM Direct Services
Sortemosevej 21
DK-3450 Allerød
Denmark

- **Fax** – send orders to:

United States (toll free)
Canada
Outside North America

1-800-445-9269
1-800-267-4455
(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States) or (+1) 408 256 5422 (Outside USA)** – ask for:

Index # 4421 Abstracts of new redbooks
Index # 4422 IBM redbooks
Index # 4420 Redbooks for last six months

- **On the World Wide Web**

Redbooks Web Site <http://www.redbooks.ibm.com>
IBM Direct Publications Catalog <http://www.elink.ibm.link.ibm.com/pbl/pbl>

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (<http://www.redbooks.ibm.com/redpieces.html>). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

Index

Symbols

./rhosts file 413

A

access control list 20, 46, 89, 90
adapter tuning 409
Address Protocol Request (ARP) 274
admin user 37, 39, 41, 56, 81
Advisors 214, 217, 282
AFS 17
 client patch for Windows NT 69
 clients 4
 file system 3, 75
 Integrated Logon 91
 on AIX 91
 on Windows NT 100
 login 93, 97
 partition 29
 port configuration 79
 security 20
 servers 4, 17
 user 85
 user authentication 90
 volume 18, 46
afs user 37
afsConfigureServer.ksh file 33
afsd daemon 44
AIX kernel extension facility 36, 53
AIX login 97
aliases configuration 257
Authentication Database 37, 39, 49, 88
Authentication Server 37, 49
authorization checking 48
automatic cache refreshing 115

B

Backup Database 39, 49
Backup Server 38, 49
backup volume 19
basic caching 158
Basic Overseer Server (BOS) 36
Best selection method 228
Binary Distribution Machine 39
boot file for AFS 49, 56
BOS Server 39
buserver 38

C

cache freshness 111
cache indexing 114
cache maintenance 108
cache management 110
Cache Manager 18, 21, 42, 54, 72
cache size 65, 114
cached data validation 19, 103

cacheinfo file 44, 54
caching 18
Caching and Filtering component 3, 5, 8, 107
caching proxy function 109
callback 19, 103
cell 17
CellServDB file 37, 42
client anonymity 108
clock synchronization on AIX 54
clock synchronization on Windows NT 77
cluster 258
cluster address 247, 257
co-location option 212
Configuration and Administration Forms 152
content backup 19
content filtering 108
customizable Advisors 213, 215

D

Database Server Machine 39
delving 115
directory tree 17
Dispatcher configuration 248
 methods 248
Dispatcher function 209, 210, 213
Dispatcher high availability 7, 211, 313
Distributed Web Traffic Express (DWTE) 8, 12
DNS 413

E

eNetwork Dispatcher (eND) 3, 6, 209
enterprise campus 349
Executor 214, 217, 252, 253
extra route 271

F

File Server 38, 49
File Server Machine 18, 39
File Sharing component 3, 17
file space 17
fileserv 38, 49
filtering function 110
firewall high availability 338
first AFS machine 22
flexible-client SOCKS 108, 117
flow of the IP packets 273
fragment size 31
fs command 40
fsck program 32

G

garbage collection 114, 162
goActive script 327
goInOp script 327
goStandby script 327

H

- header information 119
- hierarchical caching 12
- high availability 210, 212
- home directory ownership 93
- home volume 19
- htadm command 178
- HTML header 183
- HTTP header 182
- httpd configuration file 149

I

- ics_pics configuration file 149
- information flow 216
- Interactive Session Support (ISS) 211, 290
 - cell 218, 222
 - configuration files 219, 295
 - function 6, 209, 218
 - high availability 7, 211, 223, 295, 298
 - observers 226
 - resource 224
 - selection methods 228
 - service 218, 224
- issd command 298

J

- Java Servlet support 149
- javelin configuration file 149

K

- kaserver 37, 49
- Kerberos 5, 20

L

- label bureau 183, 194
- large ISP 399
- last save wins 103
- Load Balancing component 3, 8, 209
- loopback device on AIX 267
- loopback device on Windows NT 268

M

- Manager 214, 217, 282
- Maximum Transmission Unit 409
- MBUFS 410
- Media Access Control (MAC) address 274
- metric 225
- mutual authentication 20

N

- Network File System (NFS) 20
- node 218, 223
- non-forwarding address 247, 254
- number of processes per user 410

P

- Page Size 65
- paging space 410
- PICS filtering at the proxy server level 184
- PICS labels 108, 181
- PICS labels in the Web documents 204
- PICS protocol 116, 181
- PICS-compliant Web browsers 185
- Platform for Internet Content Selection (PICS) 6, 107, 180
- port 260
- ports used by the Dispatcher 217
- proportions of importance 215, 285
- Protection Database 38, 39, 49, 87
- Protection Server 38, 49
- Proxy Activity Monitor 108, 119
- proxy chaining 120, 164
- proxy function 153
- proxy server access protection 119
- proxy server protection 174
- ptserver 38, 49
- pure proxy function 157

R

- RAT file 189
- rating service 182
- rating system 181
- rc.afsd.large file 44, 54
- rc.afsd.medium file 44, 54
- rc.afsd.small file 44, 54
- ReadOnly 18
- ReadOnly replica 47
- ReadOnly volume 47
- recovery strategy 333
- regional and local access ISP 367
- regular mount point 46
- Remote Caching Access (RCA) 8
- replica 18
- Replication 18
- root volume 39
- round-robin 209
- round-robin load balancing 275
- RoundRobin selection method 228
- rule-based load balancing 213, 300
- runntp 55

S

- Salvager 38, 49
- salvager 38, 49
- server affinity 212
- server failure 228
- server machines 17
- server processes 17
- servlet configuration file 149
- smoothing index 286
- socks configuration file 149
- space quota 89
- SSL tunneling 120
- stand-alone AFS client on AIX 52
- stand-alone AFS client on Windows NT 58

stand-alone AFS server machine 50
Stat Entries 65
System Control Machine 39

T

TCP servers configuration 267
TCP/IP tuning 409
tcp_recvspace 409
tcp_sendspace 409
ThisCell file 37, 42
token 48

U

upclient 39, 49
Update Server 39, 49
upserver 39, 49
URL identifiers 184
user's volume 88
user's volume mount point 88

V

viserver 38, 49
volserver 38, 49
volume 18
Volume Location Database 38, 39, 49
Volume Location Server 38, 49
Volume Server 38, 49

W

Web Traffic Express (WTE) 3
WebSphere Application Server 15
WebSphere Studio 16
wide area support 213

ITSO Redbook Evaluation

IBM WebSphere Performance Pack Usage and Administration
SG24-5233-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at <http://www.redbooks.ibm.com>
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?

Customer **Business Partner** **Solution Developer** **IBM employee**
 None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes___ No___

If no, please explain:

What other redbooks would you like to see published?

Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)

SG24-5233-00

Printed in the U.S.A.

