

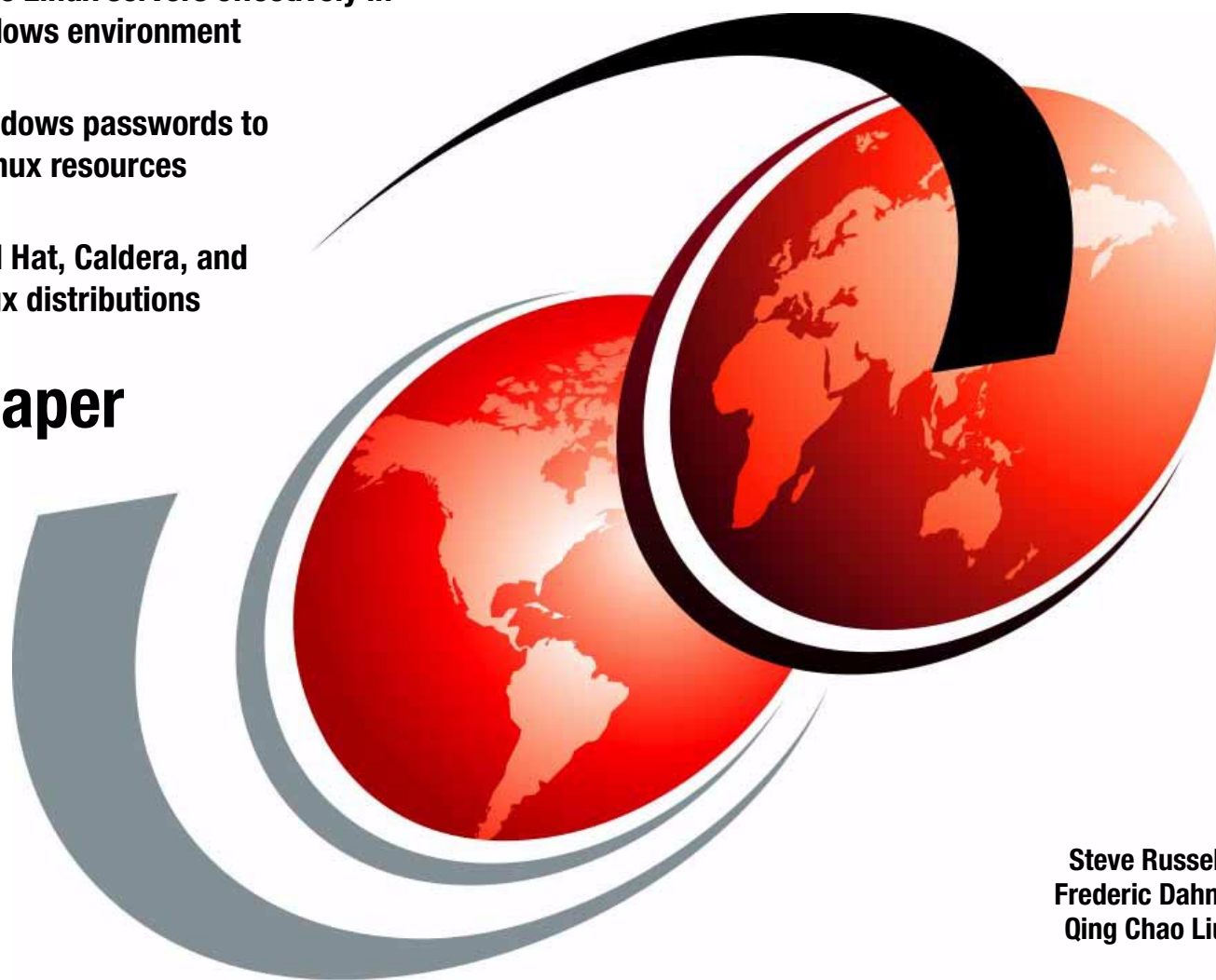
Linux and Windows Integration for IBM server xSeries Servers

How to use Linux servers effectively in
your Windows environment

Using Windows passwords to
access Linux resources

SuSE, Red Hat, Caldera, and
TurboLinux distributions

Redpaper



Steve Russell
Frederic Dahm
Qing Chao Liu



International Technical Support Organization

**Linux and Windows Integration for IBM @server
xSeries Servers**

November 2001

Take Note! Before using this information and the product it supports, be sure to read the general information in “Special notices” on page 93.

First Edition (November 2001)

This edition applies to IBM @server xSeries servers running the SuSE Linux V7.2, Red Hat Linux V7.1, Caldera OpenLinux V3.1, and TurboLinux Server V6.5 operating systems.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 662
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 2001. 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

Contents	iii
Preface	vii
The team that wrote this Redpaper	vii
Notice	viii
IBM trademarks	viii
Comments welcome	ix
Chapter 1. Introduction	1
1.1 IBM xSeries servers	1
1.2 What do we mean by “integration”?	2
1.3 The IBM commitment to Linux	2
1.4 Software referenced in this Redpaper	3
1.5 SMB and CIFS	3
Chapter 2. Technical basics	5
2.1 IBM xSeries servers	5
2.2 Linux	6
2.2.1 Linux daemons	7
2.3 Windows	10
2.3.1 Windows services, domains and Active Directory overview	10
2.4 Networking technologies	11
2.4.1 TCP/IP basics	12
2.4.2 Networking in a NetBIOS world	15
2.4.3 Network File System on Linux	18
Chapter 3. Operating system and Samba installation	19
3.1 Our reference implementation	20
3.1.1 The IBM xSeries servers used for this Redpaper	20
3.1.2 Hardware considerations	21
3.1.3 Making the CD-ROM drive bootable	22
3.2 Linux operating system installations	23
3.2.1 SuSE Linux 7.2 installation	23
3.2.2 Red Hat Linux 7.1 installation	26
3.2.3 Caldera OpenLinux Server 3.1 Installation	27
3.2.4 TurboLinux Server 6.5 Installation	31
3.3 Windows 2000 Server installation	34
3.3.1 Installing Windows 2000 Server on xSeries servers	34
3.4 Installing Samba 2.2.1a on Linux servers	36
3.4.1 Installing Samba on SuSE Linux 7.2	36
3.4.2 Installing Samba on Red Hat Linux 7.1	38
3.4.3 Installing Samba on Caldera OpenLinux 3.1	39
3.4.4 Installing Samba on TurboLinux Server 6.5	40
Chapter 4. Linux and Windows integration	43
4.1 File and print services	43
4.1.1 File and print sharing on Windows	44
4.1.2 File and print sharing on Linux	44
4.1.3 Integration considerations	44

4.2	Basic SMB connections	44
4.2.1	Creating connections from Linux using smbclient	45
4.2.2	Mounting SMB shares with the mount command	46
4.3	Printing between Linux and Windows 2000	47
4.4	Setting up the SMB Server on Linux	49
4.4.1	Starting the SMB service automatically	49
4.4.2	Configuring the Samba Web Admin Tool (SWAT)	49
4.4.3	Migrating user accounts for Samba	50
4.4.4	Configuring global settings for Samba	50
4.5	Setting up the SMB server on SuSE Linux 7.2	52
4.5.1	Starting the SMB service during the boot process	53
4.5.2	Configuring SWAT on SuSE Linux 7.2	53
4.5.3	Migrating user accounts for Samba	55
4.5.4	Configuring Samba global settings	55
4.5.5	Using SWAT to configure the Samba server	55
4.5.6	Creating shares	58
4.5.7	Working with Domain level security	58
4.6	Setting up the SMB Server on Red Hat Linux 7.1	59
4.6.1	Starting the SMB service during the boot process	60
4.6.2	Configuring SWAT on Red Hat Linux 7.1	61
4.6.3	Migrating user accounts for Samba	63
4.6.4	Configuring Samba global settings	63
4.6.5	Using SWAT to configure the Samba server	63
4.6.6	Creating shares	66
4.6.7	Working with domain level security	66
4.7	Setting up the SMB Server on Caldera OpenLinux 3.1	67
4.7.1	Starting the SMB service during the boot process	68
4.7.2	Configuring SWAT on Caldera OpenLinux 3.1	69
4.7.3	Migrating user accounts for Samba	70
4.7.4	Configuring Samba global settings	71
4.7.5	Using SWAT to configure the Samba server	71
4.7.6	Creating shares	73
4.7.7	Working with Domain level security	73
4.8	Setting up the SMB Server on TurboLinux Server 6.5	74
4.8.1	Starting the SMB service during the boot process	75
4.8.2	Configuring SWAT on TurboLinux 6.5	76
4.8.3	Migrating user accounts for Samba	77
4.8.4	Configuring Samba global settings	78
4.8.5	Using SWAT to configure the Samba server	78
4.8.6	Creating shares	80
4.8.7	Working with domain level security	80
4.9	Troubleshooting the integration process	81
4.9.1	Common network problems	82
4.9.2	Troubleshooting Samba-specific problems	82
4.10	Managing the integrated infrastructure	83
4.10.1	User account creation and management	83
4.10.2	ACL and user name mapping	84
	Chapter 5. Users	85
5.1	Browsing My Network Places	85
5.2	Adding shares to Network Places	87
5.3	Mapping network drives	88
5.4	Accessing Linux shares published in the Active Directory	89

Appendix A. Useful references 91
Web sites 91
Referenced Redbooks 92
Special notices 93

Preface

In the world of information technology, where things evolve at a rapid and quickening pace, we have seen both Linux and Windows develop considerably over the last few years. The new Version 2.4 Linux kernel, with its increased performance, scalability and flexibility, has brought increased acceptance of Linux as a server platform, as well as providing additional ease of use for clients making Linux their operating system of choice. Manifest improvements in Windows, with the introduction of Windows 2000 in particular, have also been evident. The Windows server back-end is more scalable, robust, and fault-tolerant, while the client provides simpler installation and increased ease of use.

Many people think that Windows 2000 and Linux are at opposite ends of the spectrum. Preferring homogeneous environments, system administrators install either Windows or Linux without really considering that the benefits of both platforms could be utilized in an implementation where they work together seamlessly.

The purpose of this Redpaper is to demonstrate how Windows 2000 and Linux can peacefully coexist in an integrated environment based on IBM @server xSeries servers, to provide excellent business services for any infrastructure. This is the case whether your servers will take their place in an existing infrastructure, or become the basis of a new networked environment that is planned to use both operating systems from the outset.

To this end, we start by reviewing the specifics of the two operating systems and IBM xSeries servers. We then outline those technical basics of the two systems necessary to properly understand the integration process, along with the networking details required for integration. After briefly covering operating system installation, we delve into the integration process, explaining how to configure the software that allows interoperation of the two very different systems. We continue by discussing maintenance and troubleshooting aspects of the integrated environment. The paper closes with a glimpse of how the integrated environment and services appear to the end user.

The focus of the paper is exclusively on those areas relevant to successfully integrating Windows and Linux in a single environment, and we assume that readers are reasonably familiar with these operating systems. There are any number of books that can help fill any gaps in your knowledge about their basic operation. Linux comes in a number of different flavors, or distributions. However, so we give the necessary information for all four distributions of Linux supported by IBM xSeries servers, namely: SuSE Linux V7.2, Red Hat Linux V7.1, Caldera OpenLinux V3.1, and TurboLinux Server V6.5.

The team that wrote this Redpaper

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



Steve Russell is a Certified Consulting IT Specialist at the International Technical Support Organization, Raleigh Center, where he manages residencies and produces Redbooks related to IBM xSeries servers. Before joining the ITSO, Steve worked in the Technical marketing field in the UK as a member of the IBM Netfinity organization in EMEA. Prior to that, he spent nearly 15 years managing and developing PC-based hardware and software projects at the IBM Hursley laboratory in the UK. He holds a degree in Electrical and Electronic Engineering, and is a member of the Institution of Electrical Engineers and a Chartered Engineer.



Frederic Dahm is a Systems Architect for Lotus Professional Services (LPS) in Zürich, Switzerland, where he undertakes infrastructure projects at client sites. Before joining LPS in Switzerland, he worked for three years as a systems engineer for Lotus and IBM in Ottawa, Canada, as part of the Canada East team. Although a generalist at heart, he has a strong affinity towards Security, Application Development and Linux. His experience with both UNIX and Windows systems spans three decades. He also co-authored the Redbook *Lotus Notes and Domino R5.0 Infrastructure Revealed*, SG24-5341 (ISBN: 0738413089).



Qing Chao Liu is an IT Specialist working at the Technical Support Center of IBM China. He holds a degree in Computer Science from the University of Science and Technology of China. He has four years experience with Linux systems and is also an MCSE for both Windows NT V4.0 and Windows 2000.

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

Paul Chengler
Nicolas Luder

IBM Raleigh, xSeries Briefing Center
Credit Suisse Private Banking

Notice


This publication is intended to help administrators who want to implement a heterogeneous environment that integrates xSeries servers running Windows 2000 Server and Linux V2.4 to provide enhanced services to their user population. The information in this publication does not purport to prefer one operating system over the other, and aims to provide an unbiased overview of the integration of the two operating systems and how this is best accomplished.

The information in this publication is not intended as the specification of any programming interfaces that are provided by any of the operating systems discussed. See the relevant product documentation for more information about what publications are considered to provide those specifications.

IBM trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

e (logo)®
Chipkill™
ClusterProven™
e (logo)®
Home Director™
IBM®
iSeries™
Netfinity®
OnForever™
OS/2®
Predictive Failure Analysis®

PS/2®
pSeries™
Redbooks Logo 
Redbooks™
ServerProven®
xSeries™
zSeries™
Lotus®
Lotus Notes®
Notes®
Domino™

Comments welcome

Your comments are important to us!

We want our Redpapers to be as helpful as possible. Send us your comments about this Redpaper or other Redbooks in one of the following ways:

- ▶ Use the online **Contact us** review redbook form found at:
ibm.com/redbooks
- ▶ Send your comments in an Internet note to:
redbook@us.ibm.com
- ▶ Mail your comments to the address on page ii.



Introduction

The goal of this Redpaper is to present relatively complex technical information regarding integrating the Linux and Windows operating systems in a comprehensible, easy-to-understand manner. By following the guidance given in this document, you will successfully create an integrated Windows and Linux environment that you can manage and troubleshoot effectively.

Prerequisites for understanding the procedures in the paper are technical knowledge of Windows 2000 Server and Linux, preferably of a distribution based on the Version 2.4 kernel. Knowledge of related areas such as TCP/IP and its suite of tools, as well as basics of the Network File System (NFS), Server Message Blocks (SMBs) and NetBIOS Message Blocks (NMBs) is also desirable.

To establish a technical baseline, we review this essential information in Chapter 2, “Technical basics” on page 5. We do not assume that the reader has knowledge of Samba, which is the primary tool used for integrating Linux and Windows and thus the main topic of this Redpaper.

1.1 IBM xSeries servers

xSeries servers are the latest Intel processor-based systems from IBM. They incorporate IBM X-architecture technology, a design blueprint that leverages existing, innovative IBM technologies to build powerful, scalable and reliable Intel processor-based servers to suit your business, whether your servers need to support 10 users or thousands.

They provide levels of reliability, performance, and manageability previously out of reach for industry-standard servers. Technologies derived from the IBM zSeries, pSeries, and iSeries servers bring mainframe class features to the industry-standard architecture found in the xSeries systems.

xSeries servers are available in the following four categories:

- ▶ Point solution servers
- ▶ Universal servers
- ▶ Rack-optimized servers
- ▶ Extremely scalable servers

For more information on the entire range of xSeries servers, please visit:

<http://www.pc.ibm.com/us/eserver/xseries>

1.2 What do we mean by “integration”?

When you consider Linux and Windows integration, two different aspects of this topic may come to mind, that is, integration at the client level and integration at the server level. Client-level integration essentially involves having the ability to install the two operating systems on different partitions of the same hard disk and thereby being able to boot up either one at will. In addition, it also involves having the ability to access files on partitions formatted in the native filesystems of each operating system no matter which of the operating systems is currently executing.

Server-level integration involves having the ability for requests from clients to be handled by one or other of the two operating systems, depending upon which of them is controlling the resource being requested. The goal is to have both operating systems cooperatively coexisting as part of a company’s comprehensive IT infrastructure.

In this Redpaper, when we talk about integration, we are specifically discussing server integration. A heterogeneous environment could be composed of machines running operating systems other than Linux 2.4 and Window 2000, but we have chosen to limit ourselves to these two because of the size of the existing installed base of Windows and the growing popularity of Linux.

The base minimum of services is for the integrated servers to provide file and print services to network clients, no matter which operating system is running on the server. This then implies achieving seamless operations and the ability for the different servers to collaborate in the delivery of services up to and including authenticating users with a common set of credentials.

1.3 The IBM commitment to Linux

IBM is fully committed to the open source movement and believes Linux will emerge as a key platform for e-business. Demonstrating this commitment, IBM works with the open-source community, bringing relevant technologies and experience to the table to help enhance Linux, to define the standards, and to extend Linux to the enterprise.

Three main areas are the focus of continued support and participation by IBM:

- ▶ The Open Source Development Lab
- ▶ IBM Development and Competency Centers for Linux
- ▶ IBM Technology Center

IBM has teamed with four leading commercial Linux distributors, namely Red Hat, SuSE, Caldera, and TurboLinux, to port, test, and certify the performance of IBM offerings running on various Linux distributions, enabling the exploitation of the full potential of Linux.

1.4 Software referenced in this Redpaper

The following distributions and releases of Linux are discussed in this Redpaper:

Table 1-1 Linux Distributions discussed in this Redpaper

Company	Release number	URL
Red Hat	Release 7.1	http://www.redhat.com
SuSE	Release 7.2 Professional Edition	http://www.suse.com
Caldera	Release 3.1, OpenLinux Server	http://www.caldera.com
TurboLinux	Release 6.5, Server Edition	http://www.turbolinux.com

Note: Other distributions and releases of Linux may be used. The specific products listed above produced the results tested and demonstrated in our lab during the writing of this Redpaper. Results as well as performance can vary with other distributions of Linux.

The following release of Windows is discussed in this Redpaper:

Table 1-2 Microsoft Windows release discussed in this Redpaper

Company	Release number	URL
Microsoft	Windows 2000 Server Edition, Service Pack 2	http://www.microsoft.com/windows2000

Note: Other releases of Windows can be used; however, the specific release above produced the results demonstrated and tested in our lab for this Redpaper. Results as well as performance can vary with other releases of Windows.

For the integration itself, we used Samba Release 2.2.1a.

Table 1-3 Samba release covered in this Redpaper

Company	Release number	URL
Samba	Release 2.2.1a	http://www.samba.com

Note: While we advocate using a distribution of Linux based on Release 2.4 of the Linux kernel, it is worth noting that there are no dependencies, insofar as Samba is concerned, on this release of the kernel. This means that you could use Release 2.2, for example.

1.5 SMB and CIFS

Microsoft originally created the Server Message Block (SMB) protocol to allow network clients to access network resources held on network servers. More recently, Microsoft offered an open-source enhanced version of SMB, called the Common Internet File System (CIFS).

It is important to note that the Server Message Block (SMB) protocol is now often referred to interchangeably with the Common Internet File System (CIFS). CIFS is intended to define a standard protocol for remote file access, specifically over the Internet. It can also be used to provide said services on corporate networks also.

However, since this Redpaper wishes to be as clear as possible in its dissertation of technical matters pertaining to integration of Linux and Windows, the authors felt it best, through the Redpaper, to use the acronym SMB as opposed to CIFS.

Linux supports SMB and CIFS through a package called Samba, which offers file and print services to SMB/CIFS clients.

For more information about CIFS, how it works and how it differs from SMB, consult the white paper titled "CIFS: A Common Internet File System" written by Paul Leach and Dan Perry, which can be found on Microsoft's Web site at the following URL:

<http://www.microsoft.com/mind/1196/cifs.htm>

For a good overview of SMB, consult the white paper titled "Just what is SMB?" written by Richard Sharpe in September 1999, which can be found at the following URL:

<http://samba.anu.edu.au/cifs/docs/what-is-smb.html>



Technical basics

Before you can contemplate running a network that integrates Windows 2000 and Linux, a minimal degree of technical knowledge is necessary, in order to understand and properly implement the software and services described in this Redpaper.

Even with this technical knowledge at hand, the concept of bringing in an alternate operating system in an existing infrastructure, or building a new infrastructure with two operating systems, can seem daunting, opening the doors to new technical difficulties, and creating an unwanted knowledge gap. This is particularly the case if you are familiar with Windows but less so with Linux.

In this chapter, with the knowledge gap created by introducing a new operating system in your existing infrastructure in mind, we provide the following basic technical information needed:

- ▶ An overview of the IBM xSeries servers
- ▶ An overview of Linux
- ▶ An overview of Windows
- ▶ An overview of network technologies, namely TCP/IP, SMB, NetBIOS and NFS

When appropriate, we refer you to other sources of information for coverage of particular topics in more detail than we are able to give in this paper.

2.1 IBM xSeries servers

The IBM xSeries family of servers are Intel processor-based systems introduced by IBM as successors to the award-winning range of Netfinity servers. These new servers combine the latest technologies with the proven track record of X-architecture benefits to provide an outstanding platform for business-critical information systems. The key features found in xSeries systems are:

- ▶ Scalability

The IBM strategy regarding scalability for xSeries servers is to extend performance and provide room for growth in a way that combines both cost efficiency and flexibility.

IBM also has developed the X-Architecture, an initiative that leverages enterprise-level technologies from its midrange and mainframe systems, borrows many years of in-depth experience in improving performance of server systems, and delivers better scalability with xSeries servers.

► **Robustness**

The IBM OnForever program is a system-level solution that provides leadership availability for xSeries servers, delivering the following functions:

- *Predictive Failure Analysis*: environmental monitoring sensors proactively monitor critical components and warn of impending problems.
- *Light Path Diagnostics*: improves availability by helping service personnel to identify specific failing components easily and quickly.
- *Redundant and hot-plug components*: includes subsystems such as power, cooling, and hard disk drives, allowing replacement of failed components while the server continues to operate.
- *Chipkill memory*: memory technology that offers 8-bit memory correction, greatly reducing downtime related to multi-bit memory errors.
- *Active PCI*: technology for hot-add expansion of PCI I/O adapters, which allows for increased scalability and avoids downtime.
- *ServerProven and ClusterProven programs*: extensive systems integration and testing certification programs to ensure complete compatibility and reliability of hardware and software.

► **Security**

IBM is well positioned to respond to the security needs of today's marketplace through its vast experience in enterprise security across multiple platforms, its broad offering of security hardware and software products, and its professional services organization.

► **Manageability**

Driven by the belief that improved manageability lowers the total cost of ownership (TCO), IBM has developed a comprehensive set of tools for the life-cycle management of a server. The latest versions of these tools (such as IBM Director) support various operating systems including Linux and Windows NT/2000.

2.2 Linux

Linux is an operating system that was initially created by Linus Torvalds. He began his work in 1991 when he made available Release 0.02 of the kernel through a newsgroup on the Internet. Development continued, driven by a loosely coupled team of programmers and Release 1.0 of the Linux kernel was made available in 1994. This open-source development process is still taking place and provided the current version of the Linux kernel, Release 2.4, early in 2001. This release includes a number of enterprise-class features not found in earlier kernels, such as support for journaled file systems and performance and stability enhancements.

The kernel's main role is to supply an interface to the hardware and peripherals. It also manages the memory and schedules processes. A lot of additional software is required to provide all of the functions expected from a complete operating system. Fortunately, open source developers, including IBM, have contributed time and effort to put all of the necessary pieces in place.

The Linux kernel version is labelled *A.B.C*, where *A* represents the major release number, *B* represents the minor release number, and *C* is simply an incremental number that represents patches and bug fixes. If *B* is an odd number, this denotes a kernel in development, which is not considered stable enough for a production environment. An even number for *B* denotes a stable kernel that is suitable for production environments.

Linux, both the kernel and associated software, is developed and distributed under the GNU General Public License (GPL) and its source code is freely available to everyone. The GPL means you can share and modify the software without being charged for it, and makes sure the software remains free of charge for everyone. For more information about GNU and the GPL, use following URL:

<http://www.linux.org/info/gnu.html>

There are hundreds of distributions of the Linux operating system available from sources all over the world. Most of them have their own feature set, and some are customized for special hardware configurations or for special purposes. Some well-known distributions are those supported by xSeries servers: Red Hat, Caldera, TurboLinux, and SuSE. Although the Linux software from these companies is freely available, they also sell retail versions that offer greater levels of support should you need it.

Most of the latest Linux distributions offer the following:

- ▶ Full 32-bit architecture (64-bit kernels are also available)
- ▶ Preemptive multitasking
- ▶ Support for multiple users and groups
- ▶ Full networking capabilities
- ▶ Protected memory
- ▶ Clustering, including load balancing and failover
- ▶ Dynamic reconfiguration
- ▶ SMP support

Linux is a fully functional, reliable and free operating system. This makes Linux attractive to IT managers and implementors and has permitted it to make serious inroads in the server software market. In particular, the Linux Web server, Apache, is the number one Web server in use on the Internet.

2.2.1 Linux daemons

Our ability to integrate Linux and Windows relies heavily on processes that are known as *daemons* (pronounced demons) in Linux and services in Windows, so it is worthwhile to take some time to examine them. Before describing what constitutes a daemon process, let us first discuss what constitutes a process.

A process is an executable piece of code, often called a program, that can be run by the operating system or its shell. When such a program is executed, it is loaded into memory along with any necessary data structures, file descriptors, and so on. The system's processor carries out the instructions that are contained within the program. Many processes are executed by the processor. These appear to be executed in parallel by the CPU rapidly switching from one process to another in quick succession.

A process may itself be constructed of a number of relatively independent threads that also each get a slice of the CPU's time. As an example, a program might have a thread that handles screen updates, while another manages serial data reception, and yet another sends output to a printer. A process may, if given appropriate rights to do so, access files or physical devices such as peripherals by making the necessary calls to the kernel.

Processes can be initiated by the user or by the operating system. In either case, the **init** program is responsible for initiating and terminating processes. **init** is itself the most famous of all system daemons.

A daemon process, then, is a process that is not associated with a user, and performs system functions, such as administration and control, network services, execution of time-dependent activities, or print services. In order for a process to be a daemon process, it must be able to continue after the user's session ends (which occurs after he logs off) and must not be associated with any user's terminal session. In short, a daemon is a program that runs in the background and is intended to have little or no user intervention.

Daemon processes are almost completely invisible to end users, but handle user requests and respond to various events and conditions. They are generally inactive and are designed to be called into service only when required.

How do you tell a daemon process from other processes? This is done by examining the process table. The **ps** command displays a list of active processes. In this list, a daemon can be identified by the fact a question mark "?" is listed as the terminal name associated with the process. This is identified in the TTY column of the process table. As you can see in the sample output of the **ps** command shown in Example 2-1, most of the listed processes are daemons.

Example 2-1 Linux process table (obtained with the ps command)

```
[root@redhat /]# ps ax
  PID TTY          STAT TIME  COMMAND
    1 ?            S     0:04  init [5]
    2 ?            SW    0:00  [keventd]
    3 ?            SW    0:00  [kswapd]
    4 ?            SW    0:00  [kreclaimd]
    5 ?            SW    0:00  [bdflush]
    6 ?            SW    0:00  [kupdated]
    7 ?            SW<   0:00  [mdrecoveryd]
  462 ?            S     0:00  syslogd -m 0
  467 ?            S     0:00  klogd -2
  481 ?            S     0:00  portmap
  496 ?            S     0:00  rpc.statd
  633 ?            S     0:00  /usr/sbin/automount --timeout 60 /misc file /etc/auto.m
  645 ?            S     0:00  /usr/sbin/atd
  660 ?            S     0:01  /usr/sbin/sshd
  680 ?            S     0:00  xinetd -stayalive -reuse -pidfile /var/run/xinetd.pid
  720 ?            S     0:00  sendmail: accepting connections
  733 ?            S     0:00  gpm -t ps/2 -m /dev/mouse
  745 ?            S     0:00  crond
  817 ?            S     0:00  xfs -droppriv -daemon
  852 tty1          S     0:00  /sbin/mingetty tty1
  854 tty3          S     0:00  /sbin/mingetty tty3
  855 tty4          S     0:00  /sbin/mingetty tty4
  856 tty5          S     0:00  /sbin/mingetty tty5
  857 tty6          S     0:00  /sbin/mingetty tty6
 1011 tty2          S     0:00  /sbin/mingetty tty2
 1014 ?            S     0:00  /usr/sbin/sshd
 1015 pts/0        S     0:00  -bash
 1076 ?            S     0:00  smbd
```

1674	?	S	0:00	nmbd -D
1680	?	S	0:00	CROND
1681	?	S	0:00	/bin/bash /usr/bin/run-parts /etc/cron.hourly
1699	pts/0	R	0:00	ps ax

While daemons can sometimes be started by users, they are usually started as part of the operating system's boot process. The scripts needed to start or stop the various system components including the daemons are contained in subdirectories of the `/etc/rc.d` directory. These subdirectories have names of the form `rc?.d`, where the “?” symbol is replaced by a single digit from 0 to 6, the digit representing a particular *run level* of the operating system.

Linux, from its UNIX heritage, uses the concept of run levels, which define the state in which the `init` command runs the system. There are eight predefined run levels, namely levels 0 through 6, and levels S and s. Only one of these run levels can be in operation at any time. Run levels 0 through 6 are defined as follows, with levels 0, 1 and 6 being reserved:

- ▶ Run level 0 halts the system (it is the equivalent of *shutdown* in Windows);
- ▶ Run level 1 is the single user maintenance mode
- ▶ Run level 2 is a limited multiuser mode, without network support
- ▶ Run level 3 is the complete multiuser mode
- ▶ Run level 4 is not used
- ▶ Run level 5 is the graphical user interface mode
- ▶ Run level 6 reboots the system (it is equivalent to *reboot* in Windows)

Run levels S and s are internal aliases for the same run level, which is a special kind of single-user mode. Run levels S and s are not meant to be used directly, but by scripts that are executed when entering run level 1. Run levels 7 through 9 are also valid, though not really documented. This is solely because traditional UNIX variants do not use them.

The `/etc/inittab` file contains the information that tells `init` at which run level to start the system and outlines the processes to be run at each run level. For more information on `init`, consult the relevant man pages for the distribution of Linux you are using.

Returning to the `/etc/rc?.d` directories, each contains two groups of scripts, those used to start the system (which have names beginning with the letter “S”, such as `S05network`, `S08syslog`, and `S11cron`) and those used to stop the system (which have names beginning with the letter “K” for “kill”, such as `K12cron`, `K15syslog`, and `K18network`).

Another way to identify daemons is by the fact that their command names end in the letter “d” (for daemon). For example, the Apache Web Server program, which provides HTTP services, is called `httpd`; the DNS server program, which provides name resolution services, is called `named`, and the line printer daemon, which provides remote print services, is called `lpd`.

Finally, to stop a service provided by a daemon, simply kill the corresponding daemon process using the `kill` command. This command is fairly simple: it takes a process ID (PID), which you obtain with the use of the `ps` command (see Example 2-1), as its input and terminates that process. For more information on the `kill` command, consult the relevant man pages for the distribution of Linux you are using.

2.3 Windows

At the time of writing this Redpaper, there is some confusion in regards to Windows. Depending upon how you look at it, Windows is either a shell, as KDE or GNOME are for Linux, or a completely integrated operating system with a graphical user interface (GUI). The reason for this confusion stems from the fact that there are two code streams for Windows, based on different kernels.

The first code stream evolved from MS-DOS, through the early versions of Windows (up to Windows 3.11) and then, with a major GUI update, to Windows 95, 98, and, finally, Windows Me. At this point this “home user” version of Windows has completely replaced MS-DOS, but its heritage is clearly visible.

The second code stream evolved from OS/2, which became Windows NT with the release of Windows NT 3.1 in 1993. This was the platform aimed at network servers, workstations and software development machines. Even though the GUI of this code stream of Windows looked similar to that of Windows 3.1, the kernel was totally different.

Windows NT 3.5 was released shortly afterwards, in 1994. This release came in two flavors, namely Windows 3.5 Workstation and Windows 3.5 Server. A point release, Windows NT 3.51, was offered in 1995. Windows NT 4.0 was released in 1996 with a GUI similar to Windows 95, but built on Windows NT's unique kernel.

Following in the change of version naming affecting Windows 95, 98 and Me, Windows 2000 was released in 2000. It was an overhaul of Windows NT 4.0 particularly in the areas of security, networking and directories.

Finally, in October 2001, Windows XP (XP is a quasi-acronym for eXPerience) was released, in which Microsoft intends to unite both code streams we have discussed into a single platform. There is also Windows CE, which is aimed at personal digital assistant (PDA) devices, but this specific variant of Windows falls outside the scope of this Redpaper.

The version of Windows that we discuss in regard to integration with Linux in this Redpaper is Windows 2000, specifically the Windows 2000 Server Edition.

2.3.1 Windows services, domains and Active Directory overview

Windows NT and 2000 servers have full networking capabilities, and a full implementation of TCP/IP services (such as DNS, DHCP, and so on), which come with the Windows system as components. Windows servers also provide some services widely used in the Windows world but that are rarely used on other systems, such as the file and printer sharing service (SMB) and the Windows Internet Name Service (WINS). Both Windows NT and Windows 2000 also provide directory services, but in different ways, which we now examine.

Windows NT 4.0 Server directory service

Windows NT 4.0 Server implements its directory service using the concept of a domain. A Windows NT Server domain is the administrative unit of Windows NT Server Directory Services (NTDS). The term *domain* does not refer to a single location or specific type of network configuration. Computers in a single domain can share physical proximity on a small local area network (LAN) or can be located at different sites, communicating over a variety of physical connections.

Within a domain, an administrator creates an account for each user. Each account includes user information, group membership details, and security policy information. Through the domain structure, Microsoft Windows NT 4.0 Directory Services provides several important and desirable features:

- ▶ *Single-user logon*

Network users can connect to multiple servers with a single network logon. Directory Services extends this logon to all Windows NT Server services and server applications belonging to the domain.

- ▶ *Centralized network administration*

A centralized view of the entire network from any workstation on the network provides the ability to track and manage information about users, groups, and resources in a distributed network. This single point of administration for multiple servers simplifies the management of a Windows NT Server-based network.

- ▶ *Universal access to resources*

One domain user account and password is all a user needs to use available resources throughout the network. Through Directory Services, account validation is extended to allow seamless user access to multiple network domains.

Within a domain, domain controllers manage all aspects of user-domain interaction. Domain controllers are computers that share one directory database to store information for the entire domain. They use the information in the directory database to authenticate users logging on to domain accounts. There are two types of domain controllers:

- ▶ The primary domain controller (PDC) tracks changes made to domain accounts. The PDC is the only domain server that receives these changes directly. A domain has one PDC.
- ▶ The backup domain controller (BDC) maintains a copy of the directory database. This copy is synchronized periodically and automatically with the PDC. Multiple BDCs can exist in a domain.

Windows 2000 Server directory services

The directory services of Windows 2000 Server, referred to as the Active Directory, is an improved version of Windows NT Directory Services. A Windows 2000 Active Directory can contain domains, trees, and forests, where a *tree* refers to a group of domains that is in a contiguous DNS name space, and a *forest* refers to a group of trees that are in a non-contiguous DNS name space.

The advantages of single user logon, centralized network administration and universal access to resources found in Windows NT are preserved, but there are no PDCs and BDCs anymore. Instead, Windows 2000 implements multiple peer domain controllers. Changes to the Active Directory database can be made on any domain controller, and are replicated to all others.

Entire books have been written about Active Directory and to go into detailed discussions about it is beyond the scope of this paper. It is mentioned on occasion as necessary, but for more information you should refer to the Microsoft Windows 2000 documentation.

2.4 Networking technologies

In the early days of personal computing, computers were *stand-alone* computers. They were not connected to other computers, except for perhaps for an occasional connection established using a modem or serial cable connection. Twenty years later, with the pervasiveness of the Internet and the need to communicate and collaborate, it is now considered almost unthinkable not to connect computers to a network, be it to the Internet or to a corporate network. In this section we discuss networking essentials, since these are crucial to enabling us to integrate Linux and Windows servers.

2.4.1 TCP/IP basics

The Transmission Control Protocol/Internet Protocol (TCP/IP) suite is the protocol of choice for the Internet and corporate networks worldwide. The most widely used protocols of this suite are the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). TCP is a connection-oriented protocol (for services such as delivering Web pages, transferring files, mail, and so on) and UDP is connectionless (for services such as streaming media). As such, by the nature of the data it transmits, UDP provides little error correction. The occasional corrupted byte in a video stream will go unnoticed, but you want to know that mail you receive contains the exact same message that was originally sent.

All network services discussed in this Redpaper use TCP/IP. In the following sections we discuss TCP/IP technologies that you should consider using (if you are not already doing so) when integrating Windows and Linux.

Dynamic Host Configuration Protocol (DHCP)

TCP/IP is based on the fact that each computer's network interface card (NIC) has a unique IP address. Before the massive growth in Internet usage we have seen in recent years, there were typically only a few computers on an IP network. In this situation, it is possible to manually define and set a static IP address (and related configuration data) on each computer. TCP/IP networks have grown rapidly, however, and defining and setting static IP addresses for every computer has become a time-consuming and error-prone task. Without careful control, it is possible for two systems to be given the same IP address, which is not allowed by the protocol, and errors and loss of connectivity occur.

DHCP was designed to solve this problem. The DHCP standard defines a way for DHCP servers to manage the dynamic allocation of IP addresses and other related configuration details in response to requests for this service from DHCP-enabled clients on the network. Currently, all Windows and Linux systems can act as either DHCP servers or DHCP clients.

The first time a DHCP-enabled client starts, it follows an initialization process to obtain an IP address lease from a DHCP server. The steps for this process are as follows:

1. The DHCP client broadcasts a DHCPDISCOVER message to the local subnet.
2. A DHCP server can respond with a DHCPOFFER message that contains an offered IP address for lease to the client.
3. If no DHCP servers respond to the client discovery request, typically the client fails to initialize. However, it continues to periodically resend DHCPDISCOVER messages in the background until it receives a DHCPOFFER message from a DHCP server.
4. As soon as a DHCPOFFER message is received, the client selects the offered address by replying to the server with a DHCP request.
5. Typically, the offering server sends a DHCPACK message to confirm the lease. Also, other DHCP option information is included in the acknowledgment.
6. Once the client receives an acknowledgment, it configures its TCP/IP properties using the information in the reply and joins the IP network.

DHCP considerations

DHCP is very helpful but sometimes can cause problems too. A client computer without an IP address at boot time will broadcast a DHCPDISCOVER message on the network. There could be a computer other than the expected DHCP server providing DHCP services. This means the client computer could receive an incorrect IP configuration, which could prevent it from joining the network, resulting in a form of denial of service (DoS) attack.

This is a common problem today, because even client systems such as Windows 98, Windows Me, and Windows 2000 Professional come with Internet Connection Sharing that can provide DHCP services. It doesn't matter which system you choose as a DHCP server, but do ensure that you prevent rogue DHCP servers from serving client computers.

Sometimes, in a Windows 2000 environment, you may need new DHCP features such as the ability to update DNS records dynamically and client class-specified configuration options. You should check the documentation for your DHCP server to verify whether these features are supported to avoid any future problems.

Much Internet-related documentation is contained in Requests for Comments (RFCs). DHCP-related information can be found at:

<http://www.ietf.org/rfc/rfc1531.txt>

The DHCP working group can be found at:

<http://www.ietf.cnri.reston.va.us/html.charters/dhc-charter.html>

Name resolution

Computers are accessed through TCP/IP by using their IP addresses. An IP address is represented as *w.x.y.z* (with *w.x.y.z* being numbers in the range 0 to 255), such as 192.168.0.1. It is hard to remember which computer has which address so, to make it easier for people, a system to access computers by name was designed.

Originally, this was enabled by creating a static table of computer names and related IP addresses in a file, known as the hosts file. As networks grew bigger and bigger, the task of maintaining this file and distributing it to all computers on the network became too difficult. The Domain Name System (DNS) was introduced to solve this problem.

DNS-related technologies include:

- ▶ A DNS domain name space, which specifies a structured hierarchy of domains used to organize names.
- ▶ Resource records, which map DNS domain names to a specific type of resource information for use when the name is registered or resolved in the name space.
- ▶ DNS servers, which store and answer name queries for resource records.
- ▶ DNS clients, also known as resolvers, which query servers to look up and resolve names to a type of resource record specified in the query.

Related RFC information can be found at:

<http://www.ietf.org/rfc/rfc1034.txt>

<http://www.ietf.org/rfc/rfc1035.txt>

The DNS domain name space

The DNS domain name space is based on the concept of a tree of named domains. Each level of the tree can represent either a branch or a leaf of the tree. A branch is a level where more than one name is used to identify a collection of named resources. A leaf represents a single name used once at that level to indicate a specific resource. Figure 2-1 shows how domains and computers are placed in the name space:

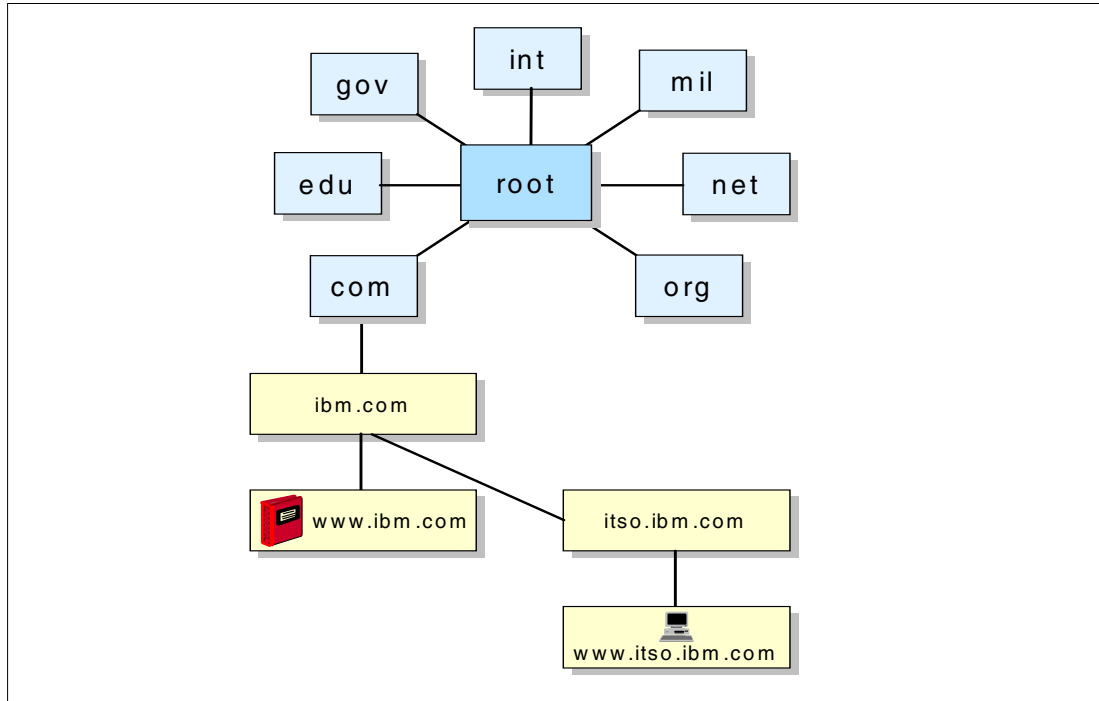


Figure 2-1 DNS name space

ibm.com and itso.ibm.com are branches of the tree, whereas www.ibm.com and www.itso.ibm.com are leaves.

DNS clients and servers use queries as the fundamental method of resolving names in the tree to specific types of resource information. This information is provided by DNS servers in response to queries made by DNS clients, who then extract the information and pass it to applications to aid them in resolving the queried name.

The DNS query

DNS queries are resolved in a number of different ways:

- ▶ A client can sometimes answer a query locally, using cached information obtained from a previous query. The DNS server can use its own cache of resource record information to answer a query.
- ▶ A DNS server can query or contact other DNS servers on behalf of the requesting client to fully resolve the name, then send an answer back to the client, in a recursive process.
- ▶ The client itself can attempt to contact additional DNS servers to resolve a name. When a client does so, it uses separate and additional queries based on referral answers from servers, in an iterative process.

So, the DNS query process thus occurs in two parts:

- ▶ A name query begins at a client computer application and is passed to a resolver, the DNS client service.
- ▶ When the query cannot be resolved locally, DNS servers can be queried as needed.

DNS replication

When the domain name space is stored in DNS servers, it is divided into zones. A zone is a group of resource records that describe the name space for domain or domains.

Because DNS is essential for today's networks, it is intended that zones be serviced by more than one DNS server on the network to provide load balancing and fault tolerance when resolving name queries. Otherwise, if a single server is used and that server is not responding, queries for names in the zone can fail. For more than one server to host a zone, a server should be configured as a primary server for this zone and other servers configured as secondary servers. Modifications can only be made on the primary server and are then replicated to the secondary servers.

DNS integration considerations

Windows NT 4.0 Server, Windows 2000 Server and Linux systems can host DNS services for name resolution. All DNS services can be configured to operate in a zone as a primary server or as a secondary server. However, Windows 2000 requires some new DNS features such as incremental zone transfer (IXFR) and dynamic update. In addition, some DNS records created by Windows 2000 Active Directory will be considered as illegal records by BIND DNS servers (that is, the type of DNS server typically used by Linux systems). So the nature of your infrastructure determines which system should be used to hold the DNS service, and how.

2.4.2 Networking in a NetBIOS world

Windows was originally built using NetBIOS as its protocol of choice, so it is important to examine its operation to better understand the integration topics covered in later chapters.

What is NetBIOS?

In 1984, IBM and Sytec co-authored a simple application programming interface (API) called Network Basic Input/Output System (NetBIOS) for networking computers in a simple LAN. It was extended in 1985 and named the NetBIOS Extended User Interface (NetBEUI).

These protocols were designed for small LANs where there is no network packet routing. As TCP/IP gained in popularity, the demand for network routing became greater, and NetBIOS over TCP/IP (or UDP/IP) was introduced. NetBIOS over TCP/IP is sometimes shortened to the acronym NBT. In 1987 the Internet Engineering Task Force (IETF) published RFCs 1001 and 1002 in an effort to standardize this protocol. These documents can be consulted on the IETF's Web site at:

<http://www.ietf.org/rfc/rfc1001.txt>
<http://www.ietf.org/rfc/rfc1002.txt>

In addition to TCP/IP, NetBIOS has also been supported over NetBEUI, IPX and DECNet protocols. NetBIOS itself defines three sets of services for maintaining connections between computers:

- ▶ Name service.

This service listens on well-known IP port 137. It is also referred to as NetBIOS browsing. On Windows desktops, the list of computers displayed in the Network Neighborhood window is created using this service.

- ▶ Datagram service.

This service listens on well-known IP port 138, and is typically not used.

- ▶ Session service.

This service listens on well-known IP port 139. It provides the file and print shares available to the network. On Windows desktops, it is accessed using the command prompt **NET USE** command or the Map Network Drive function in Windows Explorer.

SMB

Microsoft used NetBIOS to create the SMB protocol. This is a higher-layer protocol that operates above NetBIOS over TCP/IP, using the International Organization for Standardization (ISO) Open Systems Interconnect (OSI) reference model shown in Figure 2-2:

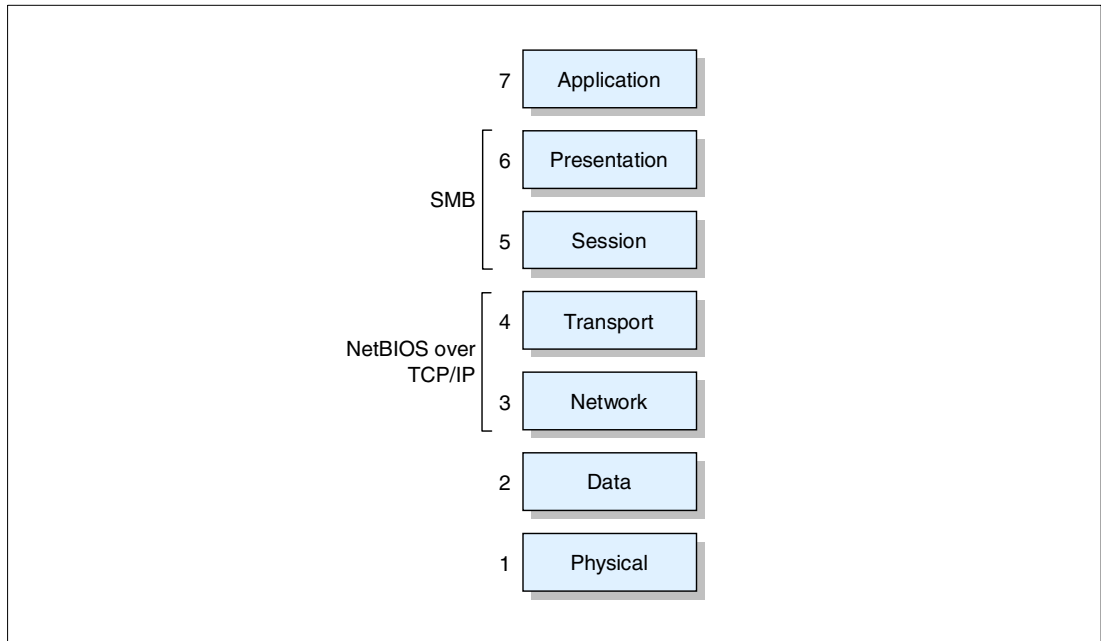


Figure 2-2 The OSI seven-layer model

NetBIOS over TCP/IP occupies the network and transport (3 and 4) layers, while SMB occupies the presentation and session (6 and 5) layers.

SMB is a client/server, request-response protocol. The specification defines two levels of security:

- ▶ *Share level.*

Protection is applied at the share level on a server. Each share can have a password, and a client needs only that password to access all files on that share. This was the first security model that SMB had.

- ▶ *User Level.*

This may be applied to individual files in a share and is based on user access rights. Each user (client) must log in to the server and be authenticated to gain access to a file. Once authenticated, the client is given a UID that it must present on all subsequent accesses to the server.

Network browsing

How can NetBIOS users find servers on the network? There is a method that uses a static table (*lmhosts*) so clients can simply be configured to know about the servers in their environment. However, changes are not automatically applied when new servers are introduced or old ones removed and, for the same reasons that DNS was developed for a TCP/IP environment, an automated solution is preferable.

To provide the answer, network browsing was introduced. Each server broadcasts information about its presence. Clients listen for these broadcasts and build browse lists. In a NetBEUI environment, this is a satisfactory solution, but in a TCP/IP environment, problems arise. The problems exist because TCP/IP broadcasts are not usually sent outside the subnet in which they originate, so cross-subnet browsing cannot be established.

To solve this limitation in TCP/IP, Microsoft introduced the Windows Internet Name Service (WINS). WINS provides a distributed database for registering and querying dynamic mappings of NetBIOS names for computers and groups used on a Windows network.

WINS basically maps NetBIOS names to IP addresses. When a WINS client starts its SMB service, it will register itself to the preconfigured WINS server. Because the WINS server is preconfigured by using an IP address or a host name, the registration can cross subnets enabling other WINS clients to find it (because they retrieve information from the WINS server) and are then able to access resources on it.

All Windows systems can act as WINS clients, with a special component called NETBT.SYS, which permits the above described name resolution. There are four such name resolution methods, configured by *Node Type*, namely B-Node, P-Node, M-Node and H-Node. These are explained in Table 2-1.

Table 2-1 NetBIOS over TCP/IP node types

Node Type	Description
B-Node	B-Node stands for <i>Broadcast Node</i> , where broadcast NetBIOS name queries are used for name registration and resolution. However, the use of B-Node could have a negative impact on larger networks.
P-Node	P-Node stands for <i>Point-to-Point Node</i> , where broadcast NetBIOS name queries are <i>not</i> used. They use instead a NetBIOS name server (equivalent to a WINS server) for name registration and resolution. If the server cannot be accessed, P-Node based name resolution will fail.
M-Node	M-Node stands for <i>Mixed Node</i> , where a B-Node and P-Node combination is used for name resolution. Basically, B-Node is used first and, if it fails, P-Node is used. This is a good mechanism because it provides fault tolerance when the NetBIOS name server is unavailable. As stated, the use of B-Node here could have a negative impact on larger networks.
H-Node	H-Node stands for <i>Hybrid Node</i> , where a B-Node and P-Node combination is used for name resolution. In contrast to M-Node, H-Node uses B-Node as a backup mechanism, when P-Node attempts have failed to access a NetBIOS Name Server. This is less taxing on larger networks because B-Node is used only as a fallback mechanism and, in order to minimize the use of B-Node, the client will continue to try to access the NetBIOS name server so that it can revert to using P-Node.

By default, Windows NT and 2000 systems use H-Node. The node type can be changed on the Windows Server by modifying the Windows Registry (using REGEDT32). The key to be changed is:

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Netbt\Parameters

It takes a number (REG_DWORD) as its value. The default value is 1 or 8, depending on the WINS server configuration, and allowed values are 1, 2, 4 and 8. They represent the following nodes:

- ▶ 1 represents B-Node
- ▶ 2 represents P-Node

- ▶ 4 represents M-Node
- ▶ 8 represents H-Node

If you do change the registry, be careful, because this can damage your server's configuration and can prevent it from working properly. To successfully integrate Linux and Windows, it is essential that all servers be configured to use NetBIOS appropriately. H-Node, the default, is therefore a good setting and should be left as is.

2.4.3 Network File System on Linux

The Network File System (NFS) was developed to allow machines to mount a disk partition on a remote machine as if it were on a local hard drive. This allows for fast, seamless sharing of files across a network.

NFS and Samba, introduced in 1.5, "SMB and CIFS" on page 3, are very similar. Both have a client and server application, both allow a server to share files with clients, and both have clients and servers available on almost every platform. The big difference is that the various flavors of Windows provide Samba-compatible clients and servers as part of their default network support, but require third-party software to support NFS. Conversely, UNIX systems usually come with and use NFS by default, and so Samba must be added to provide file sharing with Windows PCs without incurring additional costs.

All Linux 2.4 kernels support the full NFS Version 3 implementation. In addition, all Linux kernels after 2.2.18 support NFS over TCP on the client side. The server-side implementation of NFS over TCP is not supported on Linux kernels prior to 2.4.

Working with NFS

As shown in Table 2-2, working with NFS is fairly straightforward, and similar in nature to mapping a network drive:

Table 2-2 NFS and mapping a network drive

Mapping a network drive	Using NFS
On the Server	
Select a folder and share it on the server	Select a directory and add the path to <code>/etc/exports</code>
Set user permissions	Edit <code>/etc/exports</code> to define which users on which client computers have what type of access (read-only or read-write) to a specific NFS share
On the Client	
Map a network share to a drive letter	Mount an NFS share to a directory
Use the drive letter to access the network share	Use the directory to access the NFS share

Windows NT 4.0 and Windows 2000 servers can act as NFS servers with some add-on software. A package sold by Microsoft, called *Services for UNIX*, offers this capability and may be useful in networks not using Active Directory, for example. Information about the current release, Release 2.0, can be found at the following Web site:

<http://www.microsoft.com/windows2000/sfu/>

While it is therefore possible to share files between Linux and Windows by using NFS, it is more convenient, and less costly, to use NetBIOS. This is the approach that we take in this Redpaper, using Samba as the main tool for that purpose.



Operating system and Samba installation

In this chapter, we focus on the installation of the software that permits us to integrate Linux and Windows servers in a mixed environment.

To begin, we briefly describe the hardware details of the network we established in our lab as a reference system during the writing of this paper. This is followed by a summary of the steps necessary to install each of the four distributions of Linux supported by IBM xSeries servers, namely:

- ▶ SuSE Linux 7.2 Professional Edition
- ▶ Red Hat Linux 7.1
- ▶ Caldera OpenLinux 3.1
- ▶ TurboLinux 6.5 Server

We recommend that you also have a look at the extensive SuSE, Red Hat, Caldera and TurboLinux manuals, which cover the installation process in more detail and cover more variations than we have time and space to describe in this Redpaper.

Brief coverage of the Windows 2000 installation process follows and we close the chapter with Samba installation for each of the Linux distributions. Once completed, this provides us with a stable and controlled environment, which we use to discuss integration of the two operating systems in Chapter 4, “Linux and Windows integration” on page 43.

Before you start the installation of the operating systems, make sure that you check the product Web sites for updates and bug fixes. This is particularly true for Linux, as it is a fast-moving target, and its development is a continuous ongoing process. There might be new boot floppy images or kernel patches that contain newer drivers. Also, make sure that you add all security fixes if you plan to connect your machine to the Internet.




3.1 Our reference implementation



The reference implementation we developed in our lab while writing this paper uses IBM xSeries servers communicating over a local area network (LAN). It comprises two Windows servers and four Linux servers.

3.1.1 The IBM xSeries servers used for this Redpaper

The IBM xSeries servers listed in Table 3-1 were configured with either Linux 2.4 or Windows 2000 Server in our lab. All members of the xSeries family of servers can act as Linux- or Windows- based servers, but these are the ones that we chose to use for this project:

Table 3-1 IBM xSeries servers used in this Redpaper

Server	Model	Description
xSeries Universal Servers		
	xSeries 220 Server (Model 8646-22X)	The IBM xSeries 220 server is powerful enough to support file-and-print serving, workgroup productivity applications and e-commerce. To maximize your investment, this server can scale up to two processors and 4 GB of memory.
	xSeries 230 Server (Model 8658-6RY)	A server for departments or workgroups, this two-way system offers high availability, excellent internal scalability and high performance in an extremely serviceable package.
xSeries Rack-Optimized Servers		
	xSeries 300 Server (Model 8672-22X)	The new, ultra-dense, rack-optimized IBM x300 can help streamline your IT infrastructure. It delivers a cost- saving, dedicated solution, built into one of the most efficient 1U designs on the market. The single processor/dual hard drive configurations are intended for edge-of-network serving.

Server	Model	Description
	xSeries 330 Server (Model 8674-31X)	The x330 is engineered to support growing Web environments. The x330 offers support for high-performance dual processing (with fast Intel Pentium III processors), memory growth up to 4 GB and sophisticated management tools to provide a simple, powerful solution.
	xSeries 340 Server (Model 8656-6RY)	This compact box is easy to use with excellent scalability and manageability, and a number of high availability features.

Complete information and specifications for these IBM xSeries servers can be found at the IBM Web site at the following URL

<http://www.pc.ibm.com/us/eserver/xseries/>

3.1.2 Hardware considerations

Before installing the operating systems, it is helpful to know the hardware components in the server that will be used for the installation. Windows 2000 and recent distributions of Linux are capable of detecting most of these components correctly. However, it pays to know exactly which components are installed in the machine in case something is not correctly identified by the installation tools.

In particular, we recommend that you determine the following information about the components that should be present in the servers on which you are installing the operating systems:

- ▶ *SCSI adapter* - manufacturer and model number
- ▶ *Hard disk drives* - interface type (SCSI or IDE) and storage capacity
- ▶ *CD-ROM* - interface type (SCSI or IDE)
- ▶ *Display Adapter* - manufacturer, model and video memory size
- ▶ *Mouse* - mouse type and connector type
- ▶ *Network card* - manufacturer and model
- ▶ *RAM* - the amount of random access memory in your system
- ▶ *CPU* - the type and number of processors
- ▶ *Monitor* - manufacturer and model, horizontal and vertical frequency range

A very helpful resource for information about IBM xSeries servers and other IBM products, including monitors and SCSI adapters, can be found at the following site:

<ftp://ftp.pc.ibm.com/pcicrse/psref>

This archive contains Personal Systems Reference (PSREF) sheets for all IBM PC products, both current and withdrawn. You can also get a lot of useful information about IBM hardware at the following Web sites:

<http://www.pc.ibm.com/support/>

<http://www.pc.ibm.com/us/eserver/xseries/library/index.html>

In addition, the operating system vendors provide sites that list hardware compatibility information for their respective operating systems. These are listed below:

Red Hat Linux Hardware Compatibility Lists:

<http://www.redhat.com/support/hardware/>

SuSE Linux Component DataBase (CDB):

http://hardwaredb.suse.de/index.php?LANG=en_US

Caldera OpenLinux Compatible Hardware:

http://wdb1.caldera.com/chwp/owa/hch_search_form

TurboLinux Certified Hardware Compatibility List:

<http://www.turboLinux.com/hcl/TTlist.html>

Latest Windows 2000/Windows NT Hardware Compatibility List (HCL):

<http://support.microsoft.com/support/kb/articles/Q131/3/03.ASP>

3.1.3 Making the CD-ROM drive bootable

If you plan on installing your operating system by booting the server directly from the CD-ROM, make sure the CD-ROM drive is the initial boot device prior to the installation. This can be accomplished by following these steps:

1. Power on the server.
2. When you see the IBM logo, press F1 to enter the Setup utility.
3. From the Setup utility, select **Start Options**.
4. From the Start Options, select **Startup Sequence**.
5. Make sure that your CD-ROM is the initial boot device.
6. Press Esc until you see the main window of the Setup utility and select **Save Settings**.
7. Press Enter to confirm saving the current settings.
8. Exit the Setup utility.

Note: Making the CD-ROM bootable can also be done by loading the default settings from the Setup utility, but be aware that all other settings will be set to default as well.

3.2 Linux operating system installations

In this section we look at the procedures required to successfully install each of the four distributions of Linux that IBM supports on xSeries servers. Linux installation used to be a task not for the faint-hearted, requiring a lot of manual steps and tailoring to successfully complete installation. Now, Linux installation is not really any more difficult than for Windows, since many conveniences familiar to Windows users, such as auto-detection of installed hardware, have been introduced.

We assume that you can boot from the CD-ROM to install. If this is not possible, you should boot from the installation diskette, but otherwise follow the same steps.

We wanted to include the installation procedures for the four distributions we cover to assist those readers not too familiar with Linux and also to illustrate the similarities and differences between the distributions. We do not, however, list all options available during the installation process, just those needed to correctly install the operating system. Please refer to your Linux distribution installation manuals for more information.

Please refer to the appropriate section listed below for your distribution:

- ▶ 3.2.1, “SuSE Linux 7.2 installation” on page 23
- ▶ 3.2.2, “Red Hat Linux 7.1 installation” on page 26
- ▶ 3.2.3, “Caldera OpenLinux Server 3.1 Installation” on page 27
- ▶ 3.2.4, “TurboLinux Server 6.5 Installation” on page 31

3.2.1 SuSE Linux 7.2 installation

SuSE Linux 7.2 Professional allows two types of installation. The first way is to use YaST1, the console-based installation. The other is to use YaST2, a GUI installer that allows a quick and easy way to install the operating system. YaST is an acronym for *Yet another Setup Tool*, with the number after it representing the version of the installer.

YaST2 is now commonly used to install SuSE and has been tested on many of the IBM xSeries servers. While there were underlying problems with the S3 video chipset in Release 7.0 of SuSE Linux (which can make installation with Yast2 infeasible), these problems were not encountered with Release 7.2.

The YaST2 GUI installer is covered in this installation of SuSE, which makes short work of installing the Linux distribution, as the instructions below show.

Booting the installation system

1. Insert the first SuSE Linux 7.2 CD-ROM in the server’s CD-ROM drive and reboot or power up the system.

After a short delay, you should see the graphical boot splash window. It has a number of options, namely to choose an automated installation or a manual installation, whether to rescue the system or to do a memory test. As well, using the function keys, it is possible to define the video format used: F2 for text mode, F3 for VGA (640 x 480) graphical mode, or F4 for SVGA (800 x 600) graphical mode. By default, if you don’t press any keys, the automated installation in VGA mode will be chosen, which is recommended.

Basic configuration

2. Select the installation language. We chose **English (US)**. Once the language has been selected, click **Next**.

3. Select the keyboard layout. We chose **English (US)**.
4. Select the the time zone. As we were in Raleigh, North Carolina, USA, we chose **US/Eastern**.

Note: You can test that the keyboard layout is correct by using the Keyboard Test Input field to test special keys.

5. Accept the default for the hardware clock setting, which is **Local time**. Then click **Next**.
6. Now select the type of installation. We chose **New Installation**. Click the **Next** button.
7. You now have to decide how to partition your system. If you have more than one disk in your server, YaST2 will prompt you to select one of those disks onto which it will install Linux. We chose the first hard disk and accepted the default partitioning from YaST2. Alternatively, you can choose to perform partitioning manually.

Note: Manual partitioning should be undertaken only by someone familiar with the system and who understands disk partitioning. If there is data already on your disk drives, it can be lost forever if partitioning is not performed correctly.

Click **Next**.

8. If you already have an existing installation on the hard disk, YaST2 will ask you which partitions you want to delete to make room for SuSE Linux. Again, someone familiar with the system should make this decision. We did not wish to retain any existing data so we selected **Use Entire hard disk**. Click **Next**.

Software selection

9. Now you need to specify which software components you wish to install. The selections are as follows:

- Minimum System

This includes just the bare essentials necessary to safely run SuSE Linux. Selecting this option eliminates all graphical desktop environments, (that is, no X11, no KDE, and no Gnome).

- Minimum Graphical System (without KDE)

This includes the SuSE Linux base system and all packages required for X11, the graphical user interface (GUI). You will thereafter be able to work with a simple graphical desktop (with “window maker” as the windows manager).

- Default System

The SuSE default system is a good software selection for most users. You will not need to insert all the CDs that come with SuSE Linux for this selection, and can install any additional software when required.

- Default System with Office

This extends the default system to include the Sun StarOffice suite of applications.

- All Packages

All software provided is installed, from daemons, to games, to Sun StarOffice, which can take quite a long time and uses a lot of hard disk space.

We selected the **Default System** installation.

10. Click **Detailed Selection**, which permits you to select additional software to install. We need to make sure that Samba is installed and it is convenient to do so now.

11. Either:

- Select the **Network/Server** checkbox, or
- Click **Select single packages...** and scroll down the list box until you reach **Networking/Daemons**, then scroll down the list of packages for that group and double-click to select the Samba package (if it is selected, this is indicated with a “+”).

Click **Next**.

System boot configuration

12. At this stage, if your system has multiple bootable partitions (if you are configuring a Windows/Linux dual-boot system, for example), YaST2 will prompt you for information to configure the LILO (Linux LOader) boot manager.

In our case, there are no other operating systems available, so YaST2 installs the LILO boot manager on the first hard disk. You also have the option to perform a custom LILO installation if required. Click **Next**.

Personalize

13. Next, you are asked for information about the first Linux user account to be created on your server. Enter the first name and last names and a user login (or let the system suggest one by clicking **Suggestion**). This login name is used to log into the system when you are not logging in as root.

Enter the password to be used by the account and enter it again (to confirm and avoid errors), and click **Next**.

14. You now have to provide a password for the root account. Provide a password and repeat it as before, then click **Next**.

15. A window will then show all of your selected installation options as a final confirmation before beginning the installation. If everything is fine, then click **Next**.

16. A final warning window is displayed, where YaST2 requests final confirmation that you wish to install Linux on the server's hard disk. Click **Yes - install** to proceed.

17. The software packages found on the first CD-ROM are now installed. A window indicating that the basic installation is finished is then displayed, and requests that the system be rebooted. Click **OK** to reboot the server. YaST2 exits its graphical mode and displays a text box, asking you to remove the CD-ROM and any floppy disks from their respective drives. Having removed any disks, press Enter and the machine is rebooted.

Completing package installation

18. The machine will boot up with the LILO window. After a brief delay, Linux is loaded. A window prompts you to insert the second CD-ROM. Click **OK**. YaST2 will then go into text mode for a while (a normal step in the installation process), configuring the kernel and loading the software. Depending on your installation options, further CDs may be required. Follow the on-screen directions until the process is complete.

Log in to KDE

19. At this stage, Linux and KDE are running and you will be prompted to log in. To ensure that the configuration is correct, enter the login name and the password you created in step 13.

Since you are not logging in as root, the SuSE greeting will be displayed, which introduces a wizard to help you configure your desktop. This is a small extra step to complete the configuration of KDE. Click **Next**.

20. Select a theme. We chose the SuSE theme for lower color screens. Click **Next**.

21. Select the SuSE menu and click **Next**.

SuSE Linux is now completely installed. The next step is to install Samba (go to 3.4.1, “Installing Samba on SuSE Linux 7.2” on page 36).

3.2.2 Red Hat Linux 7.1 installation

The installer for Red Hat allows you to select either a text-based or a GUI-based installation routine. Since the graphical installation tool is a little more user friendly, we use it to install this Linux distribution. Here are the steps required.

Booting the installation system

1. To begin the installation, insert the first Red Hat 7.1 CD-ROM in the server's CD-ROM drive and reboot or power up the system.

After a short amount of time you should see the boot: prompt. Press Enter to continue.

2. Select the installation language. We chose **English (US)**. Once the language has been selected, click **Next**.

Basic configuration

3. Select the keyboard model. We accepted the default, which is **Generic 105-key (International) PC**.
4. Select the keyboard layout. We chose **English (US)**.
5. Select the dead key setting you want. We accepted the default of **Enable dead keys**. Click **Next**.
6. Select the mouse type for your system. We selected **2 Button Mouse (PS/2)** under Generic and checked the option to **Emulate 3 Buttons**. Click **Next**.
7. You are now at the Welcome to Red Hat Linux window. Click **Next** and select the type of installation you want. We chose **Custom System**. Click **Next**.
8. The Disk Partitioning window is displayed. Select the option to manually partition the hard disk with Disk Druid. Click **Next** to display the Disk Druid window.
9. For the purpose of this installation, we decided to create a 6.4 GB Linux native partition with the mount point set to “/” and a 127 MB Linux swap partition. Click **Next**.

Note: Disk partitioning allows a lot of flexibility in the way you organize the data in your Linux system. We selected a simple configuration, which is not necessarily the optimal way to organize your disk space.

10. Accept the default settings to format the Linux native partition and click **Next**.
11. You are now asked where you would like the Linux Loader (LILO) to be installed. We selected the Master Boot Record (MBR). Click **Next**.

Network configuration

12. The next window asks you to specify the networking parameters for the server. Since TCP/IP is the default protocol, you will need to provide a static IP address and related information. It is accepted practice to give servers a static IP address to ensure the address is retained after being rebooted. Click **Next**.
13. The firewall configuration window is now displayed. We chose **No firewall**, because the default firewall settings have the undesirable side effect of disabling all standard TCP/IP services. We need these services to be running and accessible to successfully integrate the Linux and Windows environments. For now, leave the firewall configuration disabled,

with the option later on to customize and enable the services offered by the server and click **Next**.

Installation settings

14. Now you are asked for the language you wish Linux to use. We accepted the default language setting of **English (USA)**. Click **Next**.
15. The time zone selection window appears. Our lab is in Raleigh, North Carolina, USA, so we chose **America/New_York**. Click **Next**.
16. You are now asked to provide the root account password. Enter the password and a second time to confirm it, then click **Next**.
17. You can now select how secure you want to make the system's account passwords. For additional security we chose **Enable MD5 passwords** and **Enable shadow passwords**. Click **Next**.
18. You are now asked to select which packages you would like to install. We selected **Everything**. Red Hat Linux does not install as many packages as some other distributions (such as SuSE 7.2) and this is therefore the simplest option but does not consume a lot of disk space unnecessarily. Click **Next**.

X configuration

19. The next step is to configure the computer's video adapter settings. The video adapter present in the computer is usually detected automatically. If it cannot be automatically detected, choose **Skip X Configuration**. Click **Next**.
20. The system now asks you for details of the monitor attached to the system to ensure that it works properly with the video adapter. Choose the appropriate model from the list of monitors. If your monitor is not listed here, accept the default settings. Click **Next**.
21. The following window allows you to customize the color depth, the screen resolution, the default desktop environment and whether the default console should be graphical or text-based. We accepted the default settings. Click **Next**.

Install packages

22. The software is now installed on the machine. It will begin to install the software found on the first CD. Just follow the on-screen instructions, inserting the second CD-ROM when prompted.

Once the installation process has completed successfully, the Congratulations window appears. Click **Exit** to reboot the computer.

Log in to GNOME

At this stage, Linux and GNOME are running and you will be prompted to log in. To ensure that the configuration is correct, you need to log in as root. At the command prompt, enter root as the account name and supply the password you chose in step 16. The system will log you in and this confirms that Red Hat Linux is now correctly installed and ready to work. The next step is to install Samba (go to 3.4.2, "Installing Samba on Red Hat Linux 7.1" on page 38).

3.2.3 Caldera OpenLinux Server 3.1 Installation

The installer for Caldera OpenLinux Server 3.1 allows you to select either a text-based or a GUI-based installation routine. Since the graphical installation tool is a little more user friendly, we use it to install this Linux distribution. We also install the KDE Graphical User Interface (GUI). Here are the steps required:

Booting the installation system

1. Insert the first Caldera OpenLinux Server 3.1 CD-ROM in the server's CD-ROM drive and reboot or power up the system.

After a short delay you should see the graphical boot splash window. It has a number of options:

- Standard Install mode (recommended)
- ISA hardware support mode
- VESA install mode
- Cautious install mode
- VGA16 install mode
- Expert install mode
- Non-graphic install mode
- Unattended install mode
- Demo install mode
- Collect hardware data

We chose **Standard Install mode**. For more information about the other modes and reasons to choose an alternate mode, please refer to the Caldera installation documentation.

If you don't press any keys, the Standard Install mode will be automatically chosen. The installation kernel will be loaded and the installation tool will go through an automatic hardware detection process. It will then invoke Lizard, the Linux Installation Wizard. What follows are the installation steps for this distribution of Linux using the Lizard installation tool.

Basic configuration

2. You are now asked to select the installation language. We chose **English**. Once you have made your choice, click **Next**.
3. In the next window, you are presented the Caldera Systems Inc. Software License Agreement. If you agree with the terms and conditions of the license agreement, select **Accept this license**. Once you have done this, the **Next** button will be enabled. Click it to continue.
4. You are now asked to configure the mouse connected to the server. We selected **PS/2** as the mouse type, and **Generic PS/2** as the model. We also checked **Emulate 3 Buttons**. You can test your mouse's operation by positioning the cursor within the **Test Mouse Here** area and by clicking the mouse buttons. Once you are satisfied that the mouse works properly, click **Next**.
5. Now you have to indicate the type of keyboard you are using. You need to provide the model and layout. We accepted the default model, which is **Generic 104-key PC keyboard**. For the layout, we chose **U.S. English**. Click **Next**.

Note: You can test that the keyboard layout is correct by using the Test here input field to test special keys.

6. Video adapter configuration is the next step. This information is important to ensure that the KDE GUI is displayed properly when loaded. Lizard attempts to detect the make and model of your graphics card during the boot phase. If for some reason Lizard was unable to do so, you have the option of clicking the **Probe** button to detect the card's clock chip and video RAM. Be aware, though, that probing sometimes hangs the server, which means restarting the installation.

For the xSeries servers that we used, Lizard successfully detected the make and model of the graphics card in the machine. Click **Next**.

7. The window displayed next asks you to identify your monitor. This is necessary since sometimes the monitor doesn't support all of the graphic modes provided by the video card and the wrong graphic mode may thus prevent the KDE graphical environment from being displayed properly.

If you are unsure of your monitor type, then select **1024x768, 60 Hz** from the Typical Monitors category. This mode is supported by most monitors. Click **Next**.

8. Now you are asked to select the video mode. Select **1024x768** and click **Test this mode** to ensure that the mode works properly. When you have selected a suitable mode, click **Next** to continue.

Install Caldera OpenLinux

9. The next window asks you to select the installation target. You can choose amongst the following options:

- Update
- Entire hard disk
- Free disk space
- Prepared partitions
- Custom (experts only)

Some of these options will be grayed out for an installation on a new system. For more information about the options above, please refer to the Caldera installation manuals for more information.

We selected **Entire hard disk**. Click **Next**.

10. The next window asks you to select the hard disk to be used for the installation. We selected the first hard disk as the target on which to install Caldera OpenLinux. After selecting the disk, click **Prepare selected disk for Linux**. Lizard displays preparing... in the **Current used for:** column replacing it with Linux when the preparation is complete. Click **Next** to continue.

Note: Be careful when choosing the target hard disk. If you have more than one hard disk and some disks contain data you don't want to lose, a slip here could mean lost data.

Software selection

11. The next step is to specify which software will be installed. The selections are as follows:

- Minimum Server

This option selects just the bare essentials necessary to get OpenLinux up and running.

- Web Server

This option includes a base installation suitable for a Web Server, including HTTP and FTP daemons.

- File and Print Server

This option includes a base installation that offers file and print services, and includes a set of packages offering UNIX and Windows file sharing and printer services.

- Network Server

This option includes a base installation that permits a multi-purpose server configuration and includes a set of packages offering services such as mail, DNS, FTP, HTTP, NFS, DHCP and most importantly Samba.

- All packages

This option includes every package that comes on the Caldera OpenLinux CD-ROMs.

We selected **Network Server**, since it affords more flexibility than the File and Print Server configuration. Click **Next** to continue.

Note: If you prefer, check the **Refine Selection...** box, which lets you select additional software to install.

Personalize

12. Now you have to provide the password to be used for the root account, and create the first user account. We noticed that mouse pointer movement was a little slow at this point.

Enter the password to be used by root and enter it again (to confirm and avoid errors). Also fill in the relevant fields for the full name of the first OpenLinux user account on this system, the user login name, and the user's password (twice). This login name is the one you will use to log into the system when not logging in as root.

Click **Add** to add the new user. OpenLinux offers you the means to add more than one user account but we created just the one account in our installation. Click **Next**.

Completing configuration of the server

13. Networking parameters are the next items to be configured. Since TCP/IP is the default protocol, you will need to provide a static IP address and related information. It is accepted practice to give servers a static IP address to ensure the address is retained after being rebooted.

Select the **Interface configured statically** radio button. Enter the IP address, the subnet mask, the gateway, and the host name of the server. Optionally enter an NIS domain. Enter the IP address of the name server (DNS) and, optionally, a backup name server. Click **Next** to continue.

14. The next window lets you decide where to locate the boot loader. We selected our primary hard disk and also selected **Write master boot record**. Click **Next**.
15. You are then presented with the Setup Modem window. We clicked **Next**, as we did not need to configure a modem.
16. Similarly, for the Setup Printer(s) window we clicked **Next**, as we did not need to configure a printer.
17. The next window asks you to set up the system's time zone. We chose **US/Eastern**. Click **Next**.
18. At this point, the Entertainment window is displayed, which allows you to play a game of solitaire. It may seem odd to have that there, but because Lizard is installing the packages in the background, you have to wait for it to complete the process before being able to complete the installation process. Unless you have other things to do while Lizard installs the packages, you may as well enjoy a break. Click **Next** either now, if you decide not to play the solitaire game, or right after you have won.
19. The installation is essentially complete now. However, you are presented with the Rescue Disk window, which enables you to create a rescue disk. This is useful if the boot record of the hard disk becomes damaged, since it will permit you to boot into Linux from the floppy

disk. This is an optional (but highly recommended) step. If you decide to do it, click **Write Disk** after placing a blank floppy disk in the drive. Remember to remove the floppy disk after it has been created.

20. Click **Finish** to boot Linux on the server.

Log in to KDE

21. At this stage, Linux and KDE are running and you will be prompted to log in. To ensure that the configuration is correct, provide the login name and the password you entered in step 12 on page 30. Caldera OpenLinux is now completely installed and ready to work. The next step is to install Samba (go to 3.4.3, “Installing Samba on Caldera OpenLinux 3.1” on page 39).

3.2.4 TurboLinux Server 6.5 Installation

The currently available version of TurboLinux for servers is TurboLinux Server 6.5. This release is based on the Linux 2.2 kernel, unlike the other distributions covered in this document, which use the 2.4 kernel. In the following section, we briefly describe the installation of TurboLinux and the KDE GUI.

Booting the installation system

1. Insert the first TurboLinux Server 6.5 CD-ROM in the server's CD-ROM drive and reboot or power up the system.

After a short delay a text-based splash window appears. TurboLinux offers both a text-based and a GUI-based installation process. To use the graphical installation tool, simply press the Enter key. If, for some reason you prefer the text-based tool, type text and press the Enter key.

2. Even if you selected the GUI tool, the next window is still text-based. You are asked to select the installation language. We chose **English**. Once the language has been selected, press the Enter key. A text message is displayed: Now loading second stage of installer, which remains on until the GUI installation tool is completely loaded.

Basic configuration

3. The Select Install Class window is displayed, offering two options, namely:

- Standard Install
- Upgrade

Select **Standard Install** and click **Next** to continue.

4. Unlike the other distributions we have examined, some TurboLinux installation windows have tabs. The window currently displayed is Configure Keyboard, which has two tabs: Easy and Advanced.

We chose the Easy tab, and selected **Generic 105-key (intl) PC** as the keyboard. When this keyboard is selected, a graphical layout of the keyboard is displayed, which should match the keyboard you are using. If not, scroll down the list and select the one that corresponds to the keyboard that you are using. Using the Advanced tab gives you more detailed choices if required.

Click **Next** to continue.

Note: You can test that the keyboard layout is correct by using the Test your selection here input field to test special keys.

5. Mouse configuration is next, Here again there are Easy and Advanced tabs. We chose the Easy tab again, and selected **PS/2 mouse** as the mouse type, **2 Button Mouse** as the model, and checked **Emulate 3 Buttons**. If you want to ensure that the selected mouse settings are working correctly, place the cursor within the Test Mouse Here area and click the mouse buttons. Once you are satisfied that the mouse works properly, click **Next**.

Prepare the hard disk

6. The Partition Disk window appears next, offering three options:
 - Automatic partitioning
 - TFDisk, which is a graphical partitioning tool
 - FDisk, which is a console-based partition management tool

We selected **Automatic partitioning**. Click **Next**.

7. The Configure Boot Loader window is displayed to allow you to configure LILO (Linux LOader) boot manager. This window also has Easy and Advanced tabs. We chose the Easy tab and checked **Create boot disk**. and the **Install LILO** checkboxes. Select the **/dev/sda Master Boot Record (MBR)** radio button to specify where to install the LILO boot record, then click **Next**.

Set up the network

8. The next window asks you to specify the networking parameters for the server. Since TCP/IP is the default protocol, you will need to provide a static IP address and related information. It is accepted practice to give servers a static IP address to ensure the address is retained after being rebooted.

Clear the **Configure using DHCP** checkbox and ensure that **Activate on boot** is checked. Enter the IP address, the subnet mask, and the host name of the server. Also, enter the gateway and Primary DNS addresses and, optionally, a secondary and tertiary DNS address. Click **Next** to continue.

9. The next window to be displayed is Configure Timezone. We configured the time zone and time settings under the Location tab. Make sure the **System clock uses UTC** checkbox is cleared. We chose **America/New_York** for our location. Use the time zone most appropriate to your location.

Ensure that the time and date are correct on the server. If not, enter the correct values in the **Current time** fields. Click **Next**.

Personalize

10. The next window to be displayed contains three tabs: Easy, Create Accounts and Authentication. We use the first two. Here you provide information about the password to be used for the root account, as well as information about the first Linux user account to be created on this server.

Select the **Easy** tab and enter a password for root. Enter it again to confirm and ensure you have typed it correctly.

11. Select the **Create Accounts** tab. This is where you create the account for the first TurboLinux user on this system, which is the one you will use to log into the system when not logging in as root. Enter a user name (a maximum of eight characters), and the password to be used by this account, repeating it to confirm and ensure it is correct. Finally, enter the full name of the person to whom this account belongs. Click **Add** to create the account. You can create additional accounts if you wish. Click **Next**.

Software Selection

12. The Software Selection window contains three tabs: Standard Selection, Custom Selection, and Settings. We used only the Standard Selection tab. Here you specify which software you want to install. The selections are as follows:

- Basic System

This includes only the basic packages required for the TurboLinux system.

- Internet Server

This includes all of the basic packages plus all packages required to create an Internet server.

- Intranet Server

This includes all of the basic packages plus all packages required to create a Web server for your intranet.

- Everything

This selection installs all packages from the Install CD, including the X window system.

Since we might need to download a newer version of Samba for TurboLinux in a format other than an RPM (and thus, will need to recompile the Samba source code), we selected **Everything** to ensure that any tool we might need is available. Click **Next**.

Completing the configuration of the server

13. In the Configure Monitor window, select the manufacturer and the specific model of monitor attached to your system from the lists provided. Click **Next**.

14. The Configure X window is displayed. This is where you configure the appearance of the KDE GUI. TurboLinux scans the machine to determine the make and model of the video card installed. If it is not correctly identified, select the appropriate make and model of the video card in your system.

Select the Desktop Color Depth, and a Desktop Resolution (size) appropriate for your display. 1024x768 is a typical selection. Click **Test this configuration** to make sure your card and monitor work properly with your selections.

15. Select your desktop manager. KDE and GNOME are available. We chose to use KDE. Click **Next**.

16. The About to Install window is displayed. Click **Next** to begin package installation. A window is displayed asking whether you wish to continue with the installation. Click **Ok** to proceed.

17. After the base installation, you are prompted as to whether you want to install additional packages or not. Check **Skip additional package installation** and click **Next**.

18. The installation is essentially complete now. However, you are presented with the Create Boot Diskette window, which enables you to create a boot disk. This is useful if the boot record of the hard disk becomes damaged, since it will permit you to boot into Linux from the floppy disk. This is an optional (but highly recommended) step. If you decide to do it, click **Write Disk** after placing a blank floppy disk in the drive. Remember to remove the floppy disk after it has been created.

19. The Congratulations window is displayed. Click **Finish**, then press Enter. This will boot the newly installed version of Linux.

Log in to KDE

20. At this stage, Linux and KDE are running and you will be prompted to log in. To ensure that the configuration is correct, provide the login name and the password you entered in step

11 on page 32. TurboLinux Server is now completely installed and ready to work. The next step is to install Samba (go to 3.4.4, “Installing Samba on TurboLinux Server 6.5” on page 40).

3.3 Windows 2000 Server installation

In this section, we describe the installation of Windows 2000 Server. This is relatively straightforward, but we wanted to include it for completeness. Doing so also gives us the opportunity to illustrate and highlight differences in the installation process from one server to the next. For this purpose, we installed the operating system on xSeries 300 and 230 servers.

3.3.1 Installing Windows 2000 Server on xSeries servers

The overall installation process for Windows 2000 Server is comparable in complexity to that of the installation of any of the Linux distributions we have discussed. The overall look and feel of Windows 2000 is somewhat more polished, as one would expect, but similar steps and choices have to be made.

Booting the installation system

1. For the xSeries 300 server, simply boot from the Windows 2000 CD and press any key when prompted to start installation.

The xSeries 230 server requires slightly different initial steps. As for the x300, boot from the Windows 2000 CD and press any key when prompted to start installation. However, you must press F6 when the blue screen appears. This allows you to add additional drivers to support the SCSI chipset for the x230. When prompted, insert the “Ultra 160 SCSI manager set for Windows 2000” diskette into the floppy disk drive. Press Enter, then press the S key. Choose **Adaptec Ultra 160 Family PCI SCSI Controller Win 2000** and press Enter to continue.

2. The Windows 2000 text-based installation tool is started and appears after a short delay. Press Enter to install the Windows 2000 Server.
3. The Windows 2000 licensing agreement appears. Carefully read the licensing agreement and choose **I Agree** by pressing the F8 function key.

Preparing the hard disk

4. Windows 2000 Setup displays the hard disk storage available on the system.

Choose or create a partition to install the Windows 2000 Server. We chose to use all of the available space on the primary hard disk, which in this case has 8 GB of storage capacity. Highlight the partition and press Enter to continue with the installation.

5. The installation tool now prompts you to choose a file system. We chose Windows 2000’s native NTFS file system. Once selected, Windows Setup will begin to format the selected partition and copy the appropriate files onto the selected partition.

The graphical installation process

6. The system now restarts and displays the graphical installation tool, called the Windows 2000 Setup Wizard. The first window of the wizard is the Welcome window. Click **Next** to continue.
7. The wizard automatically installs and configures devices in the computer. This process will take several minutes.

8. After some time, the Regional Settings window is displayed, defaulting to U.S. settings. We accepted the default settings, as we were in Raleigh, North Carolina, USA. Click **Next** to continue.

Personalize

9. Setup displays the Personalize Your Software window. In the Name field, type the user name, and in the Organization field, type your organization's name. The names selected will be used as default computer names later on in the installation process.
10. Windows, unlike Linux, requires you to provide a CD key before allowing you to continue with the rest of the installation. When the wizard prompts you to enter the Windows 2000 Setup CD key, enter the key included with the installation CD in the appropriate fields. Once the CD key has been properly entered, click **Next**.
11. The wizard then displays the Licensing Modes window, offering two choices of licensing modes. Select the appropriate licensing mode for your organization, then click **Next** to continue.
12. The System Name and Administrator Password window appears. Enter the system name and the password for the Administrator account (equivalent to root in Linux). Click **Next** to continue.

Software components

13. The software component window is displayed. Decide which components you wish to be included as part of the installation process. To install a component, select the box next to its name. Components that you do not select will not be installed. Click **Next** to continue.
14. The Time Zone window is displayed. Set the current time for Windows 2000. Click **Next** to continue.

Network settings

15. When the Networking Settings window appears, choose between **Typical** or **Custom** settings. When prompted, enter the appropriate network information. We accepted the default settings. Click **Next** to install the Windows networking components.
16. Next, the Workgroup or Computer Domain window is displayed. Select one of the following options:
 - No, this computer is not on a network, it is a network without a domain.
 - Yes, make this computer a member of the following domain box.Enter the workgroup or domain name in the appropriate field if required. We chose not to join a domain during the installation process. Click **Next** to continue.

Performing the remainder of the installation

17. The wizard displays the Installing Component window, and displays the status as it installs and configures the remaining operating system components according to the options specified. This step takes several minutes.
18. Eventually, the wizard displays the Performing Final Tasks window, which shows the status as setup finishes copying the appropriate files. Remove the Windows Server CD from the CD-ROM drive and click **Finish**. The server reboots and runs the newly installed Windows 2000 Server.

Log in to Windows

19. At this stage, Windows is running and you will be prompted to log in. Press Ctrl+Alt+Delete and log in as Administrator, entering the password you assigned in

step 12 on page 35. When the desktop appears, the Windows 2000 Configure your Server Wizard will be displayed. This wizard will assist in the setup of any additional Windows 2000 components on your server.

Installing the Advanced System Management Processor driver

Note: Steps 20 through 23 are only for installations on an xSeries 230 server, or other xSeries servers that include an integrated management processor. The x300 does not have a management processor as standard, and therefore does not require these steps.

20. Open Device Manager, and double-click **Unknown Device**.
21. Select Reinstall Driver and click **Next**.
22. Select Floppy only and click **Next**.
23. Click **Next** and then click **Finish**. The Windows 2000 Server will reboot.

Installing Active Directory

24. After rebooting, the Configure Your Server Wizard will appear. Click **Active Directory** and follow the on-screen instructions to promote this Windows 2000 server to a domain controller.

Windows is now fully loaded and configured. We are now ready to integrate it with Linux, as described in Chapter 4, “Linux and Windows integration” on page 43.

3.4 Installing Samba 2.2.1a on Linux servers

Release 2.2.1a was the current version of Samba at the time we wrote this Redpaper. While each of the Linux distributions covered in this paper are shipped with Samba, all of them except SuSE have versions prior to Release 2.2. We recommend that you download the latest release for installation on your Linux servers. The RPM packages and latest release .tar and .gz files can be downloaded from:

<http://www.samba.org>

This section outlines the installation process for each of the distributions we have been discussing. Samba configuration, which is the crux of the integration mechanism between Linux and Windows, is covered in Chapter 4, “Linux and Windows integration” on page 43.

3.4.1 Installing Samba on SuSE Linux 7.2

These instructions will help you if you decide to add Samba to an existing SuSE 7.2 installation or in case you did not completely follow the SuSE Linux 7.2 installation instructions provided in 3.2.1, “SuSE Linux 7.2 installation” on page 23.

At the time of writing, Samba 2.2.1a was not available as a binary package for SuSE Linux 7.2 from the Samba Web site. If you absolutely want or need to have the latest version, you could download the relevant code and build the Samba executables. We felt Release 2.2, shipped as standard with SuSE 7.2, was recent enough and thus avoided the complication of having to compile the code.

In 3.4.4, “Installing Samba on TurboLinux Server 6.5” on page 40 we have to compile the Samba source code because a recent version of Samba is not shipped with that distribution, nor is a recent RPM package available from the Samba Web site. If you decide to build the code for SuSE, you can follow the steps given for TurboLinux.

Here are the steps required to install Samba 2.2 on SuSE 7.2:

1. To start, log in as root and determine whether or not the Samba package is installed on your system. This is done by issuing the command:

```
rpm -qa | grep samba
```

If Samba is not installed, the command will complete without displaying any output. If an older release of Samba is installed, the output of the command looks something like this:

```
suse:~ # rpm -qa | grep samba
samba-2.0.7-36
```

If a full installation of a previous version of Samba is present, additional packages will also be reported, such as:

```
suse:~ # rpm -qa | grep samba
samba-2.0.7-36
samba-common-2.0.7-36
samba-swat-2.0.7-36
samba-client-2.0.7-36
```

2. If present, you must remove the previous release of Samba before installing the current version. Do this by issuing the following commands:

```
suse:~ # rpm -e samba-2.0.7-36
suse:~ # rpm -e samba-common-2.0.7-36
suse:~ # rpm -e samba-swat-2.0.7-36
suse:~ # rpm -e samba-client-2.0.7-36
```

As indicated, you need to issue a separate command to remove each of the components found in step 1. You must, of course, substitute the exact names of the packages found on your system.

3. To install Samba from the SuSE 7.2 CDs, insert the second CD into the CD-ROM drive. Make a mount directory for the CD-ROM drive in the /root/mnt directory by issuing the following command:

```
suse:~ # mkdir /root/mnt/cdrom
```

4. Then mount the CD-ROM drive in order to be able to access it, by issuing the following command:

```
suse:~ # mount /dev/cdrom /root/mnt/cdrom
```

5. Access the CD-ROM and go to the SuSE network directory (suse/n2), by issuing the following command:

```
suse:~ # cd /root/mnt/cdrom/suse/n2
```

6. The package we wish to install is called samba.rpm. Install the package using the rpm command like this:

```
suse:~/mnt/cdrom/suse/n2 # rpm -i samba.rpm
```

7. If everything proceeds correctly, the following information is displayed:

```
Updating etc/rc.config...
/etc/inet.conf is up to date
/etc/services is up to date
```

8. At this stage, do one more small check and re-issue the rpm -qa | grep samba command, which should produce the following output:

```
suse:~/mnt/cdrom/suse/n2 # rpm -qa | grep samba
samba-2.2.0-15
```

If so, then congratulations! You have just installed Samba 2.2 on your SuSE Linux server.

3.4.2 Installing Samba on Red Hat Linux 7.1

Red Hat Linux 7.1 comes with a copy of Samba Release 2.0.7. Release 2.2.1a is more reliable, more secure, and incorporates useful new features, so we recommend that you use this new version (or a later version, if available).

This means you will not be able to conveniently install Samba at the same time as the operating system, but it generally pays to have the most up-to-date copy of the software. New features often ease the administrative burden and provide better services to your user population. We also wanted to gain experience with the latest release on our servers in our lab, and validate the infrastructure services provided.

Installing Samba 2.2.1a on Red Hat Linux 7.1 is a fairly simple and straightforward procedure, the directions for which are provided below:

1. First, log in as root, create a directory named samba, and change to that directory:

```
[root@redhat /root]# mkdir /samba
[root@redhat /samba]# cd /samba
```

2. Next, you need to download the Samba package (RPM). Go to the Samba site (<http://www.samba.org>) and click the download server closest to you. We chose one of the USA servers. Once connected to the FTP server, go to the /Binary_Packages/redhat/RPMS/7.1 directory and download samba-2.2.1a-20010717rh71.i386.rpm to the samba directory.

Note: You are likely to find a later version of Samba available by the time you read this paper. We recommend you obtain the latest release.

3. Before installing the Samba package, determine whether or not an older Samba package is already installed on your system. To do so, use the `rpm` command shown below. If you have a full installation of Samba, you will see following results:

```
[root@redhat /samba]# rpm -qa|grep samba
samba-swat-2.0.7-36
samba-common-2.0.7-36
samba-2.0.7-36
samba-client-2.0.7-36
```

4. If found, remove the previous release of Samba with following commands:

```
[root@redhat /samba]# rpm -e samba-swat-2.0.7-36
[root@redhat /samba]# rpm -e samba-client-2.0.7-36
[root@redhat /samba]# rpm -e samba-2.0.7-36
[root@redhat /samba]# rpm -e samba-common-2.0.7-36
```

As indicated, you need to issue a separate command to remove each of the components found in step 3. You must, of course, substitute the exact names of the packages found on your system.

5. Now that we have ensured that your server has no prior Samba installation on it, we can continue. Install the Samba package with this command:

```
[root@redhat /samba]# rpm -i samba-2.2.1a-20010717rh71.i386.rpm
```

6. If everything proceeds correctly, the following information is displayed:

```
Looking for old /etc/smb.conf...
Looking for old /etc/smbusers...
Looking for old /etc/lmhosts...
Looking for old /etc/MACHINE.SID...
Looking for old /etc/smbpasswd...
Moving tdb files in /var/lock/samba/*.tdb to /var/cache/samba/*.tdb
Installing stack version of /etc/pam.d/samba...
```

If so, then congratulations! You have just installed Samba 2.2.1a on your Red Hat Linux server.

3.4.3 Installing Samba on Caldera OpenLinux 3.1

Caldera OpenLinux 3.1 comes with a copy of Samba Release 2.0.8. Release 2.2.1a is more reliable, more secure, and incorporates useful new features, so we recommend that you use this new version (or a later version, if available).

This means you will not be able to conveniently install Samba at the same time as the operating system, but it generally pays to have the most up-to-date copy of the software. New features often ease the administrative burden and provide better services to your user population. We also wanted to gain experience with the latest release on our servers in our lab, and validate the infrastructure services provided.

Installing Samba 2.2.1a on Caldera OpenLinux 3.1 is a fairly simple and straightforward procedure, the directions for which are provided below:

1. First, log in as root, create a directory named samba, and change to that directory:

```
[root@caldera /root]# mkdir /samba
[root@caldera /samba]# cd /samba
```

2. Next, you need to download the Samba package (RPM). Go to the Samba site (<http://www.samba.org>) and click the download server closest to you. We chose one of the USA servers. Once connected to the FTP server, go to the /Binary_Packages/Caldera/OpenLinux/RPMS/Server-3.1/ directory, download all of the Samba 2.2.1a binary RPMs, and save them to the samba directory.

Note: You are likely to find a later version of Samba available by the time you read this paper. We recommend you obtain the latest release.

3. Before installing the Samba package, determine whether or not an older Samba package is already installed on your system. To do so, use the `rpm` command shown below. If you have a full installation of Samba, you will see following results:

```
[root@caldera /samba]# rpm -qa|grep samba
samba-2.0.8-1
[root@caldera /samba]# rpm -qa|grep smb
smbfs-2.0.8-1
```

4. If found, remove the previous release of Samba with following commands:

```
[root@caldera /root]# rpm -e smbfs-2.0.8-1
[root@caldera /root]# rpm -e samba-2.0.8-1
```

As indicated, you need to issue a separate command to remove each of the components found in step 3. You must, of course, substitute the exact names of the packages found on your system.

5. Now that we have ensured that your server has no prior Samba installation on it, we can continue. Install Samba with these commands:

```
[root@caldera samba]# rpm -i smbfs-2.2.1a-20010712.i386.rpm
[root@caldera samba]# rpm -i samba-2.2.1a-20010712.i386.rpm
[root@caldera samba]# rpm -i samba-doc-2.2.1a-20010712.i386.rpm
[root@caldera samba]# rpm -i swat-2.2.1a-20010712.i386.rpm
```

6. If everything proceeds correctly, the following information is displayed:

```
Reloading INET configuration: inetd.
```

If so, then congratulations! You have just installed Samba 2.2.1a on your Caldera OpenLinux server.

3.4.4 Installing Samba on TurboLinux Server 6.5

TurboLinux Server 6.5 comes with a copy of Samba Release 2.0.7. Release 2.2.1a is more reliable, more secure, and incorporates useful new features, so we recommend that you use this new version (or a later version, if available).

This means you will not be able to conveniently install Samba at the same time as the operating system, but it generally pays to have the most up-to-date copy of the software. New features often ease the administrative burden and provide better services to your user population. We also wanted to gain experience with the latest release on our servers in our lab, and validate the infrastructure services provided.

Installing Samba 2.2.1a on TurboLinux Server 6.5 is a little more complex than for the other distributions, but still not difficult. This is because there is no RPM package available from the Samba Web site, which means the code has to be built from the source. Directions are provided below:

1. First, login as root, create a directory named samba, and change to that directory:

```
[root@turbo /root]# mkdir /samba
[root@turbo /samba]# cd /samba
```

2. Next, you need to download the Samba package (RPM). Go to the Samba site (<http://www.samba.org>) and click on the download server closest to you. We chose one of the USA servers. Once connected to the FTP server, download the Samba source package from the root directory (we downloaded samba-2.2.1a.tar.gz) and save it to the samba directory.

Note: You are likely to find a later version of Samba available by the time you read this paper. We recommend you obtain the latest release.

3. Before installing the Samba package, determine whether or not an older Samba package is already installed on your system. To do so, use the `rpm` command shown below. If you have a full installation of Samba, you will see following results:

```
[root@turbo /samba]# rpm -qa|grep samba
samba-debugtools-2.0.7-11
samba-common-2.0.7-11
samba-2.0.7-11
samba-client-2.0.7-11
```

4. If found, remove the previous release of Samba with following commands:

```
[root@turbo /samba]# rpm -e samba-client-2.0.7-11
[root@turbo /samba]# rpm -e samba-debugtools-2.0.7-11
[root@turbo /samba]# rpm -e samba-2.0.7-11
[root@turbo /samba]# rpm -e samba-common-2.0.7-11
```

You may receive some error messages in the last step, but the packages will be removed properly.

5. Now that we have ensured that your server has no prior Samba installation on it, we can continue. To do so, switch to the samba directory and extract the Samba source files from the package with the command:

```
[root@turbo /samba]# tar -zxvf samba-2.2.1a.tar.gz
```

6. Switch to the source directory, and configure the package by entering:

```
[root@turbo /samba]# cd samba-2.2.1a/source  
[root@turbo /source]# ./configure --with-pam
```

For the next few seconds, messages will scroll past on the console as the configuration files are created.

7. The next step is to create the executables with the **make** command and then to install them in their appropriate directories:

```
[root@turbo /source]# make  
[root@turbo /source]# make install
```

Note: The first **make** command may take several minutes to complete. Messages will scroll past on the console as the Samba code is recompiled.

8. Now you have to create the PAM configuration file `/etc/pam.d/samba`, containing the following lines:

```
auth    required    /lib/security/pam_pwdb.so nullok shadow  
account required    /lib/security/pam_pwdb.so
```

9. The last step is to add the path for the Samba programs to root's profile. Edit the file `/root/.bash_profile`, to add a line before the line starting with `ENV` like this:

```
PATH=/usr/local/samba/bin:$PATH
```

10. Reboot your server

That completes the installation of Samba 2.2.1a on your TurboLinux 6.5 system.

Now that we have Samba successfully installed, we are ready to move on to the discussion of how to integrate Linux and Windows in your server environment.



Linux and Windows integration

At this point, we have our Windows and Linux servers up and running and connected to your network. Samba has been installed on the Linux servers. Now we have to make these systems work together so that users can access server resources as easily as possible, no matter which operating system is hosting those resources. This chapter explains how to integrate Windows and Linux in this way.

After a brief introduction to file and print services, we look at quick and easy ways to establish ad hoc server message block (SMB) connections from both Linux servers and clients to Windows servers in order to exchange files and information. The tools used for this do not require Samba to be configured beforehand.

Then we explain how to properly configure Samba on your Linux servers, outlining the different configuration parameters, what they do, and recommend optimal values. We cover the installation and configuration of SWAT, the Samba Web Admin Tool, and demonstrate how Samba can participate as a file and print server for Windows and Linux clients.

It is important, finally, to note that configurations and functionality vary from one Linux distribution to the next. This means the installation and configuration of Samba is not exactly the same for each distribution. To make sure that the steps are properly outlined for the distribution you are using, we provide details for each of the four distributions discussed in Chapter 3, “Operating system and Samba installation” on page 19 in their own sections. By reviewing the relevant section, all of the information you need can be found in one place, rather than being spread throughout sections of this chapter.

4.1 File and print services

Before getting into the details of integration, it is useful to review the core file and print services available in Linux and Windows.

4.1.1 File and print sharing on Windows

All Windows versions released since, and including, Windows for Workgroups provide SMB-based file and print sharing services as part of the standard configuration. With these services, users can browse their network to find out what resources are available. Those resources can then be directly accessed, either through tools such as Network Neighborhood, or by mapping a file share to a drive letter, allowing access to the share as if it were a local hard disk. For example, a user can access a network share named share1 on a server named server1, defining it as drive Z: by using the following command:

```
net use z: \\server1\share1
```

Windows 2000 also allows you to publish individual file shares in the Active Directory, making them easier to locate for end users. Support for print sharing is also very good in Windows 2000. Client printer drivers for multiple releases of Windows may be stored on the server and installed automatically as required so that clients can print to the printer without having to manually install the correct printer driver. NFS and UNIX printing services can also be supported by Windows 2000 Server.

As all current Windows systems are native SMB clients, the SMB model is almost invariably the choice for file and print sharing between Windows systems.

4.1.2 File and print sharing on Linux

Linux can provide a wide-range of support for file and print sharing, too. As a variant of UNIX, Linux includes a complete implementation of NFS for file sharing. Administrators can mount a network share to a local directory, and this is transparent to end users. For example, you can mount a directory named /share1 located on a server named server1 to the server or client's local directory named /mnt/dir1 by using the following command:

```
mount server1:/share1 /mnt/dir1
```

The printing subsystem of Linux can support remote client printing requests, with clients able to send print jobs directly to the Linux server. Linux also supports Samba, a fully functional SMB server, which provides file and print sharing for SMB clients.

As a client Linux can utilize NFS, can access UNIX printing services, and can also use SMB services. With Linux, you can have the full flexibility of choosing whichever service model you desire.

4.1.3 Integration considerations

The goal of our integration efforts is to create systems that are both easy to use for your end users, and easy to manage for your administrators. Recognizing the fact that most client systems are likely to be running Windows, it is convenient to use the SMB file sharing model. We thus use Samba on Linux to provide file sharing services. Using the native support for remote printing on Linux, and the flexibility of Windows 2000 printing services, we can make them work together to provide the best services to client computers. In the following sections we explain in detail how to configure these services.

4.2 Basic SMB connections

In the spirit of walking before running, let's look at creating basic connections for file sharing or printing from a Linux server or client to a Windows server. We examine how to establish these connections to a Windows server from the various distributions of Linux.

We assume that the appropriate release of Samba has been installed on the Linux server or client that you are using, as discussed in the instructions for each distribution in 3.4, “Installing Samba 2.2.1a on Linux servers” on page 36. The topics in this section apply to all four distributions we have discussed.

4.2.1 Creating connections from Linux using smbclient

The need to create a connection from a Linux server or client to a Windows server occurs on a regular basis in a mixed environment. The Windows server has files that are needed by the user of the Linux system, and so a way to access the files is required.

There are a couple of ways to establish such connections, which we might call *ad hoc*, since they are impromptu and generally temporary. The user establishes the connection, transfers the files and then kills the link (or the link is killed when the user logs out or restarts the Linux server or client).

The first way to do this is to use the **smbclient** command. This command comes with the Samba suite of utilities and provides FTP-like functionality over SMB. With the **smbclient** command, the user at the Linux server or client can do a number of things including transferring files to and from the Windows server, retrieving directory information, and even connecting to a shared printer on the Windows server.

The following sections give some examples of the **smbclient** command in action.

Browsing a list of shares on a specific server

If you cannot remember the share name you wish to access, it is useful to obtain a list of available shares from the Windows server. This is accomplished with the help of the **-L** parameter of the **smbclient** command as shown in Example 4-1 :

Example 4-1 Using the -L switch

```
linux:~ # smbclient -L w2k-300 -U administrator
added interface ip=192.168.0.4 bcast=192.168.0.255 nmask=255.255.255.0
Password:
Domain=[REDPAPER] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
```

Sharename	Type	Comment
-----	----	-----
cc	Disk	
IPC\$	IPC	Remote IPC
ADMIN\$	Disk	Remote Admin
C\$	Disk	Default share

Server	Comment
-----	-----
CALDERA	Samba Server on Caldera OpenLinux
W2K-230	
W2K-300	

Workgroup	Master
-----	-----
REDPAPER	W2K-230

The server’s name, shares, domains and workgroups are listed. If you are familiar with Windows, you can see that the output of the **smbclient** command with the **-L** switch provides similar information to the Windows **NET VIEW** command.

Connect to an SMB share on a Windows server

Once you know the proper name for the share, you can connect to it and use it for the purpose it was intended. This is accomplished the following command:

```
smbclient //servername/sharename password -U username
```

Where `servername` is the name of the Windows server, and `sharename` is the name of the resource to which you want to connect.

Note: Samba uses “/” as the Universal Naming Convention (UNC) path delimiter. This differs from the syntax used by Windows, which uses “\”, as in: \\servername\sharename.

The user name and password parameters are for the Windows account that you want to use in order to access the share. The output of the command looks like this when connecting to a share:

```
added interface ip=192.168.0.4 bcast=192.168.0.255 nmask=255.255.255.0
Domain=[REDPAPER] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \> _
```

The output and functionality of the `smbclient` command, used as described, is a mixture of FTP and the Windows **NET USE** command.

Among other things, you can use:

- ▶ The `ls` command to list the contents of the newly shared directory.
- ▶ The `cd` command to change directories in the share.
- ▶ The `get` command to copy files to the current local directory.
- ▶ The `lcd` command to change the current local directory.

For more information on the `smbclient` command, please consult the `smbclient` man pages that come with your distribution of Linux.

4.2.2 Mounting SMB shares with the mount command

All files accessible in Linux (as in all UNIX systems) are arranged in one big tree: the file hierarchy rooted at `/`. These files can be spread out over several devices, be it one or more hard disks, a CD-ROM drive, a floppy disk or even, when using the `smbfs` file system, on disks located on a Windows server.

The `mount` command serves to attach the file system found on a device (be it local or remote) to the file tree. The counterpart of the `mount` command is `umount`, which detaches what was attached to the tree with the `mount` command.

To mount an SMB share on a Windows server, use the following command:

```
mount -t smbfs -o username=<username>,password=<password> //servername/sharename mountpoint
```

Where `username` and `password` are the details for a valid Windows (not Linux) account, `servername` is the name of the Windows server, `sharename` is the name of the resource to which you want to connect, and `mountpoint` should be an empty local directory, such as `/mnt/smbshare`. After the share has been mounted, you can access it as a branch beneath the mount point as you would a local directory. This method is similar to the mapping of network drives in Windows.

Example 4-2 shows the result of using the `mount` command, which is independent of the distribution of Linux you use, to connect to the `C:\` drive of a Windows 2000 server:

Example 4-2 Mounting an SMB share

```
linux:/mnt # mount -t smbfs -o username=administrator //w2k-300/cc /mnt/samba
Password:
linux:/mnt # ls samba
.                IO.SYS           WINNT           pagefile.sys
..              MSDOS.SYS       arcldr.exe     ttemp23
AUTOEXEC.BAT    NTDETECT.COM    arcsetup.exe
CONFIG.SYS      Program Files    boot.ini
Documents and Settings System Volume Information ntldr
```

You can use the two above methods to transfer data between a Windows system and a Linux system. If you compare this data transfer method to other mainstream methods (such as FTP, HTTP or NFS), using the Linux system as an SMB client is easier and more convenient.

4.3 Printing between Linux and Windows 2000

There are lots of method of printing from Linux to Windows, and conversely, from Windows to Linux. In this section we introduce two methods.

In addition to its other capabilities, the `smbclient` command can connect directly to an SMB-shared printer and print files. For example, if we want to print a text file called `file.txt` to a shared Windows printer named `IBM40375` on server `w2k-4000r` (in other words, it is addressable from the network as `\\w2k-4000r\ibm40375`), we can use the commands shown in Example 4-3:

Example 4-3 Printing from a Linux client to a Windows printer

```
[root@redhat /root]# smbclient //w2k-4000r/ibm40375 -U administrator
added interface ip=9.24.105.138 bcast=9.24.105.255 nmask=255.255.255.0
Password:
Domain=[REDPAPER] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \> print file.txt
putting file file.txt as file.txt (10.3 kb/s) (average 10.3 kb/s)
```

When this command completes, `file.txt` is sent to the Windows printer.

Making Linux act as a print server in a Windows 2000 environment is even easier. Printing is one of the standard services implemented on Linux systems. As we mentioned earlier, services are generally provided by daemons. The daemon that provides print server services is called the line printer daemon (LPD), and supports local print jobs and remote print jobs submitted using TCP/IP as the transfer protocol. In addition, Windows 2000 can act as a client of LPD, which means that you can easily print from a Windows 2000 machine to a Linux server equipped with a printer.

To allow a Windows 2000 server or client to print to a Linux server, you need to:

1. Make sure your Linux is running LPD and that the printer is configured to use the proper port.
2. Install the Print Services for UNIX on Windows 2000:
 - a. Open the **Control Panel**.
 - b. Double-click **Add/Remove Programs**, then select **Add/Remove Windows Components**.
 - c. Highlight **Other Network File and Print Services**, then click **Details...**
 - d. Check the **Print Services for Unix** checkbox, as shown in Figure 4-1.

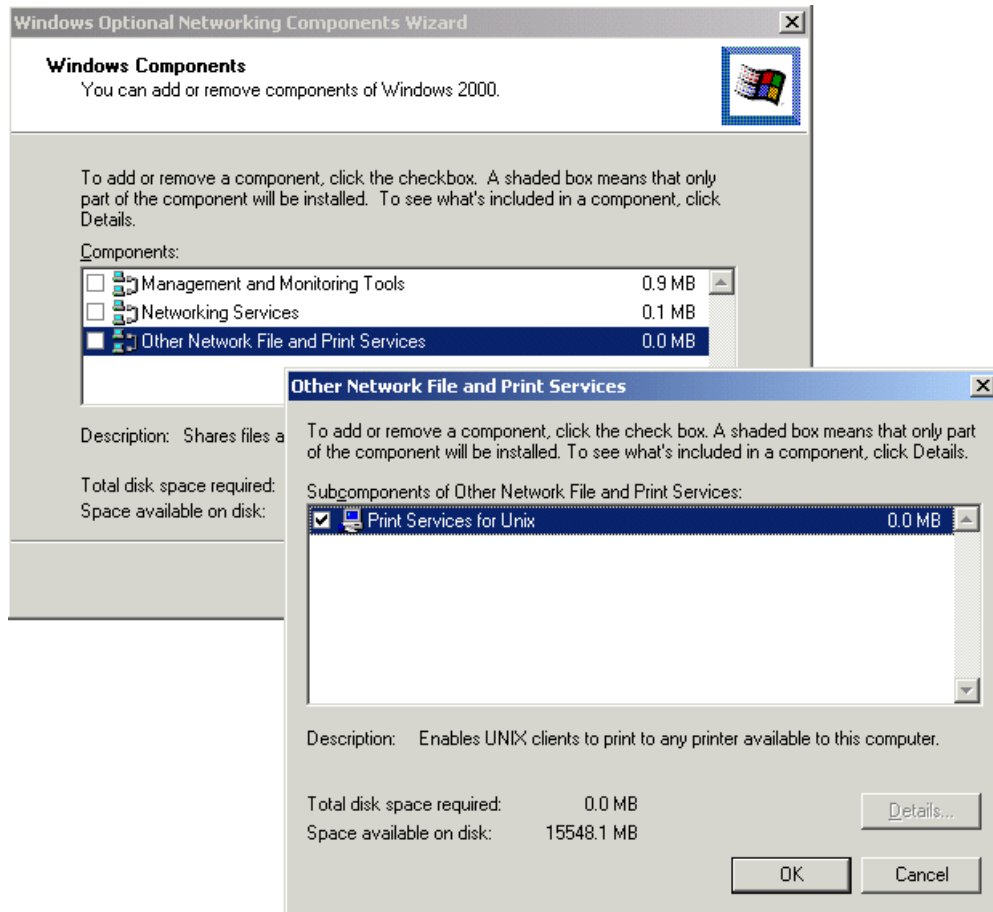


Figure 4-1 Installing Print Services for UNIX

- e. Click **OK**, then click **Next**.
 - f. Click **Close** to close the Add/Remove Programs window.
3. Add the Linux-based printer to your Windows 2000 computer:
 - a. Back in the Control Panel, double-click **Printers**.
 - b. Double-click **Add Printer**. This will start the Add Printer Wizard. Click **Next**.
 - c. Select **Local printer**, make sure that the **Automatically detect and install my plug and play printer** check box is deselected, and then click **Next**.
 - d. Select **Create a new port**, and then select **LPR Port** in the listbox. Click **Next**.
 - e. Fill in the following information:
 - In the Name or address field of the server providing LPD services, enter the Domain Name System (DNS) name or the Internet Protocol (IP) address of the server for the printer you are adding.
 - In the Name of printer or print queue field for that server, type the name of the printer as defined on the Linux server.
 - f. Complete the installation of the printer by following the instructions provided by the Wizard.

Because you added the Linux printer as a local printer, you can share it with other Windows computers and, on a Windows 2000 server, you can load drivers for client computers, so that users can print to the Linux printer without having to manually install drivers.

4.4 Setting up the SMB Server on Linux

Now we examine how to establish a more complete set of services based on SMB. Since there are tangible differences from one distribution of Linux to another, we first cover general basics, moving on to distribution-specific details in the following sections:

- ▶ 4.5, “Setting up the SMB server on SuSE Linux 7.2” on page 52
- ▶ 4.6, “Setting up the SMB Server on Red Hat Linux 7.1” on page 59
- ▶ 4.7, “Setting up the SMB Server on Caldera OpenLinux 3.1” on page 67
- ▶ 4.8, “Setting up the SMB Server on TurboLinux Server 6.5” on page 74

4.4.1 Starting the SMB service automatically

As part of the system services, the SMB service should be configured to start automatically upon system bootup. Please refer to the section that is specific to your distribution for instructions on how to do so.

4.4.2 Configuring the Samba Web Admin Tool (SWAT)

When the SMB service starts automatically upon bootup, it opens the Samba configuration file (`/etc/samba/smb.conf`) to load the Samba settings. This file contains numerous parameters, which can make it hard to properly edit and maintain.

To make the process of configuring the Samba settings (and therefore the `smb.conf` file) easier, Samba comes with a Web-based configuration tool called the Samba Web Admin Tool (SWAT). This tool enables administrators to modify `smb.conf` settings and perform tasks such as starting and stopping the SMB service using a GUI interface (something required every time any settings in the `smb.conf` file are changed).

Important: If you are working with manually created `smb.conf`, using SWAT will overwrite the `smb.conf` file, which can lead to manually set parameters being changed without your knowledge, causing some unexpected results.

SWAT is accessed using a standard Web browser. Since the interface is simple, any HTTP 1.1 compliant browser will do, such as Netscape and Opera, or if you want to access SWAT from a Windows machine, you can also use Internet Explorer. On the Linux server, SWAT itself provides all required services, including the HTTP service to serve the Web browser. This means that Apache does not need to be installed or running for SWAT to be used. All that is required is that you access the server using the SWAT TCP port, which is usually port 901.

SWAT services

When working with Linux systems, there are two ways to configure programs such as SWAT to provide services.

The first way is to run the program in the background as a daemon that listens for incoming requests from clients on specific IP ports. When a request arrives on one of those ports, the program takes the necessary actions to properly handle the request. Using this method, you need to have one process running in the background for each service you want the server to provide. Since today's servers are powerful and often expected to provide many services to client workstations, this method can be quite resource-intensive and not the optimal configuration for the server.

A second, more efficient, way to provide these services uses Internet Service Management (ISM). This is based on the fact that, typically, relatively few services (from all those that the server can provide) are requested to run simultaneously by client workstations. Given this fact, a mechanism was introduced to manage how and when services are started on the server. The basic idea is to build a table that maps ports to services and then to run a special daemon that listens to all ports used by the services it supports. When a request is made against one of the server's ports, the special daemon checks the table to find out which service is being requested by the client workstation. Then this daemon will start the service-related program and pass the request to it. This is the function of the `inetd` and `xinetd` daemons.

By default, SWAT is not started upon system bootup. It is started by daemons such as `inetd` and `xinetd`, which are sometimes referred to as *super servers*. We provide more details on how ISM is implemented, the specifics of the daemons that invoke it, and SWAT configuration in the distribution-specific sections that follow.

Configuring SWAT properly is the most complex part of enabling Samba. With the help of the section specific to your Linux distribution, you should be able to make short work of it.

4.4.3 Migrating user accounts for Samba

Before users can access SMB services provided by a Linux server, Samba-specific accounts must be established for them. Details of how to do this are also covered in the section specific to each distribution.

4.4.4 Configuring global settings for Samba

Global settings are a collection of attributes that define the behavior of Samba and the manner in which it delivers its services to SMB clients. We'll provide a list of the most important attributes, which will be used, in turn, to build a simple Samba server.

- ▶ Workgroup

This attribute controls which workgroup your server will appear to be in when queried by clients. Note that this parameter also controls the Domain name used with the `security = domain` setting.

- ▶ NetBIOS name

The attribute sets the NetBIOS name of the SMB server. It is the same as the computer name in Windows.

- ▶ Server string

The attribute is a description of the server, which can be viewed by the client by using the `net view` command. It is the same as the Description field in a Windows network.

- ▶ Interfaces

This attribute allows you to override the default network interfaces list that Samba uses for browsing. By default, Samba queries the kernel for the list of all active interfaces and uses any broadcast-capable interfaces, except 127.0.0.1 (the IP loopback address).

This option takes a list of interface strings. Each string can be in any of the following forms:

- A network interface name (such as `eth0`). This may include shell-like wildcards, so `eth*` will match any interface starting with the substring `eth`.
- An IP address. In this case the netmask is determined from the list of interfaces obtained from the kernel.
- An IP/mask pair, such as `192.168.0.1/24` or `192.168.0.1/255.255.255.0`.

► **Encrypt passwords**

This attribute defines whether or not passwords should be encrypted. In order to communicate with Windows 98, Windows Me, Windows NT, and Windows 2000, this attribute must be set to `yes`.

► **Guest account**

This attribute represents a user name that will be used for access to shares that are specified as `guest ok`. Typically, this user will exist in the password file, but will not have a valid login. The user account `ftp` is often a good choice for this parameter. If a user name is specified in a given service, the specified user name overrides this one.

► **Security**

This attribute affects how clients respond to Samba and is one of the most important settings in the configuration file (`smb.conf`). The available values are:

- `security = share`

When clients connect to a share level security server, they need not log in to the server with a valid user name and password before attempting to connect to a shared resource (although modern clients such as Windows 98, Windows Me, Windows NT, and Windows 2000 will send a logon request with a user name but no password when talking to a `security = share` server). Instead, the clients send authentication information (passwords) on a per-share basis, at the time they attempt to connect to that share.

- `security = user`

This is the default security setting in Samba 2.2. With user-level security, a client must first log in with a valid user name and password. Encrypted passwords can also be used in this security mode.

- `security = server`

In this mode Samba will try to validate the user name and password by passing the information to another SMB server, such as a Windows NT or Windows 2000 server. If this fails it will revert to `security = user`.

- `security = domain`

In this mode, Samba will try to validate the user name and password by passing it to a Windows NT primary or backup domain controller, in exactly the same way that a Windows NT or Windows 2000 server would do.

► **Host allow**

This attribute represents the hosts (clients) allowed to access shares on the server. The value of this attribute can be set in a number of different ways. For example:

```
hosts allow = 192.168. EXCEPT 192.168.0.1
allow all clients of 192.168.x.x except 192.168.0.1.
hosts allow = 192.168.0.0/255.255.255.0
```

These allow hosts that match the given network and netmask to access shares on this server. Alternatively this can be set in the following manner:

```
hosts allow = client1, client2
```

This allows two clients to access the shares on this server.

- ▶ Host deny

This attribute represents hosts (clients) that are not allowed to access shares on this server. The format of this attribute is the same as that for host allow.

- ▶ Log file

This attribute specifies the log file.

- ▶ Max log size

This attribute is an integer that specifies the maximum size (in kilobytes) to which the log file is permitted to grow. Samba periodically checks the size of the log file and, if it exceeds the specified size, Samba will rename the log file, adding a .old extension to it.

- ▶ Preferred master

This attribute indicates whether or not the Samba server should be the browsing master. The default setting is auto, which is perfectly suitable for most situations.

- ▶ WINS support

This boolean attribute controls whether or not Samba will act as a WINS server. This attribute should not be set to true unless there is a multi-subnetted network and a particular Network Message Block daemon (nmbd) is required to be the WINS server.

Note: This should *never* be set to true on more than one machine in the network.

- ▶ WINS server

This attribute specifies the IP address of the WINS server with which Samba should register. If there is a WINS server on the network, then this should be set to the WINS server's IP address.

Note: The DNS name of the WINS server can also be used. However, we recommend that you use the IP address.

4.5 Setting up the SMB server on SuSE Linux 7.2

In this section, we build a simple Samba server to provide network clients with the requisite SMB services.

In the configuration we describe, we use the following settings:

- ▶ User level security
- ▶ Allow all hosts
- ▶ Encrypted passwords

The first step in configuring our Samba server is to make sure that the SMB service starts correctly during the boot process. Next, we configure SWAT for this distribution of Linux. Finally, we use SWAT to configure the Samba server to provide the network services for our clients.

4.5.1 Starting the SMB service during the boot process

As a system service, the SMB service should be configured to start during the boot process. In order to know how to modify system settings to start a service when Linux boots, let us take a look at its boot process.

When your server is powered on or restarted, the machine's power-on self test (POST) and basic input/output system (BIOS) check and initialize your system. Once this is complete, the server boots from the hard disk and, as for all other Linux distributions covered in this Redpaper, the Linux Boot Manager (LILO) is loaded.

LILO allows you to select bootup options if you wish, but if you do nothing Linux is selected and the kernel is automatically loaded and decompressed in the server's memory. The kernel then initializes device drivers and file systems, and calls a special program, named `init`.

It is from this point that things differ among Linux distributions.

SuSE Linux 7.2 uses a tool named `SuSEconfig` to control the `init` process. Controlling which services are started during the boot process is managed by editing the file `/etc/rc.config`. To start the SMB service, you need run both the Network Message Block daemon (`nmbd`), which is the daemon responsible for NetBIOS name resolution, and the Server Message Block daemon (`smbd`), which is responsible for fulfilling the file and print service requests of SMB clients. These daemons are started during boot process by adding the following line to `/etc/rc.config`:

```
START_SMB="yes"
```

Once this is done, save the file, run `SuSEconfig`, and restart your server. When the server is running again, use the `ps` command to verify the result. The SMB service has started properly if you see the output as shown here:

```
suse:~ # ps ax|grep smbd
1076 ?      S        0:00 smbd
```

The SMB service has started as a default system service and your server has now the basic ability to act as an SMB server. There is still some extra work that needs to be done. The next step will be to configure the Samba Web Admin Tool (SWAT).

4.5.2 Configuring SWAT on SuSE Linux 7.2

Every time the SMB service starts, it opens its configuration file (`/etc/samba/smb.config`) to load the settings it contains. These settings cover all aspects of the SMB service. As we have discussed earlier, Samba comes with a Web-based configuration tool named SWAT, which is accessed using a Web browser through port 901. SWAT itself processes the Web requests (that is, it does not require Apache to run).

It is important to note that the SWAT process is not started when your Linux server boots up. Instead, a daemon such as `inetd` or `xinetd` has the responsibility for starting it. With SuSE Linux 7.2, `inetd` performs that function. `/etc/inetd.conf` is the default configuration file for the `inetd` daemon. This file enables you to specify the daemons to start by default and to supply the arguments that correspond to the required mode of operation for each daemon.

Each line in this file is of the form:

```
ServiceName SocketType ProtocolName Wait/NoWait UserName ServerPath ServerArgs
```

These fields must be delimited by spaces or tabs, and their meaning is as follows:

- ▶ *ServiceName* contains the name of an Internet service defined in the `/etc/services` file.

- ▶ *SocketType* contains the name for the type of socket used for the service. Possible values for the *SocketType* parameter are:
 - *stream* specifies that a stream socket is used for the service.
 - *dgram* specifies that a datagram socket is used for the service.
 - *sunrpc_tcp* specifies that a Sun remote procedure call (RPC) socket is used for the service, over a stream connection.
 - *sunrpc_udp* specifies that a Sun RPC socket is used for the service, over a datagram connection.
- ▶ *ProtocolName* contains the name of an Internet protocol defined in the `/etc/protocols` file. For example, use the `tcp` value for a service that uses TCP/IP and the `udp` value for a service that uses the User Datagram Protocol (UDP).
- ▶ *Wait/NoWait* contains either the `wait` or the `nowait` instruction for datagram sockets and the `nowait` instruction for stream sockets. The *Wait/NoWait* field determines whether the `inetd` daemon waits for a datagram server to release the socket before continuing to listen at the socket.
- ▶ *UserName* specifies the user name that the `inetd` daemon should use to start the server. This variable allows a server to be given less permission than the root user.
- ▶ *ServerPath* specifies the full path name of the server that the `inetd` daemon should execute to provide the service.
- ▶ *ServerArgs* specifies the command line arguments that the `inetd` daemon should use to execute the server. The maximum number of arguments is five. The first argument specifies the name of the server used. If the *SocketType* parameter is `sunrpc_tcp` or `sunrpc_udp`, the second argument specifies the program name and the third argument specifies the Release of the program. For services that the `inetd` daemon provides internally, this field should be empty.

To configure SWAT, follow these steps:

1. Edit `/etc/inetd.conf` to include this line:

```
swat stream tcp nowait.400 root /usr/sbin/swat swat
```

and save the file.

2. Now you need to make `inetd` start during the boot process. Edit `/etc/rc.config` and add this line:

```
START_INETD="yes"
```

Then save the file.

3. Run **SuSEconfig** and reboot your server.

Now that this done, you can test your SWAT settings using the following method:

4. Use `lynx` (a text browser) on the Linux server.
 - a. Switch to a text-based terminal (Linux permits you to switch between terminals by pressing `Alt+F1` through `Alt+F6` for a text mode terminal or `Alt+Ctrl+F1` through `Alt+Ctrl+F6` for a graphic mode terminal).
 - b. Enter `lynx localhost:901` and wait for several seconds. You have successfully configured SWAT if you see the following message appear:

```
Username for 'SWAT' at server 'localhost:901':
```

- c. Now you can switch to graphic mode and use a Web browser to access SWAT at:

```
http://localhost:901
```

5. If you want another workstation, specifically one running Windows, to be able to access the SWAT tool, you need edit the entry for SWAT in the file `/etc/hosts.deny`:

```
swat:ALL EXCEPT 127.0.0.1 w2k-300
```

In this example, we are allowing localhost and w2k-300 to access SWAT.

- If you do not have a DNS system that supports forward and reverse lookups for both computers, add the host names to the hosts file. The hosts file is located in the `/etc` directory on SuSE Linux 7.2 systems, `\windows` on Windows 98/Me systems and `\winnt\system32\drivers\etc\` on Windows NT/2000 systems.

Once you have done so, you can direct your browser to <http://sambaservername:901>. If you see a window requesting a user name and a password, then your configuration is successful. Otherwise, make sure you can access SWAT from the Linux server first, then check the configuration file.

4.5.3 Migrating user accounts for Samba

Samba-specific accounts must be set up for your users before they can access the SMB services now provided by your Linux server. These accounts (and related information) are held in a file called `smbpasswd`.

Initially, the `smbpasswd` file does not exist and must be created using the `smbpasswd` command. When executed, the command will prompt you for the new Samba password and create a `smbpasswd` file.

It is also possible to create an encrypted `smbpasswd` file based on the `/etc/passwd` file by using the following command:

```
suse:~ #samba# cat /etc/passwd | mksmbpasswd.sh > /etc/samba/smbpasswd
```

This ensures every person who has an account on the Linux server has a Samba account as well.

Note: Users who have been given a Samba account should change their own SMB password using the `smbpasswd` command.

4.5.4 Configuring Samba global settings

Global settings are a collection of attributes that define the behavior of Samba and the manner in which it delivers its services to SMB clients. Please consult 4.4.4, “Configuring global settings for Samba” on page 50 for a list of commonly used attributes.

4.5.5 Using SWAT to configure the Samba server

As was explained in 4.4.2, “Configuring the Samba Web Admin Tool (SWAT)” on page 49, Samba configuration is managed using a file called `/etc/samba/smb.conf`. You can modify this with a normal text editor or by using the Samba Web Admin Tool (SWAT), which, as its name implies, provides access through a Web browser. This is the tool we chose to use as Windows users are generally more comfortable with a GUI interface. However, remember that using SWAT may change settings you may have made manually, so it is best to choose one method and use that exclusively.

Open your preferred Web browser and connect to SWAT as shown in Figure 4-2:

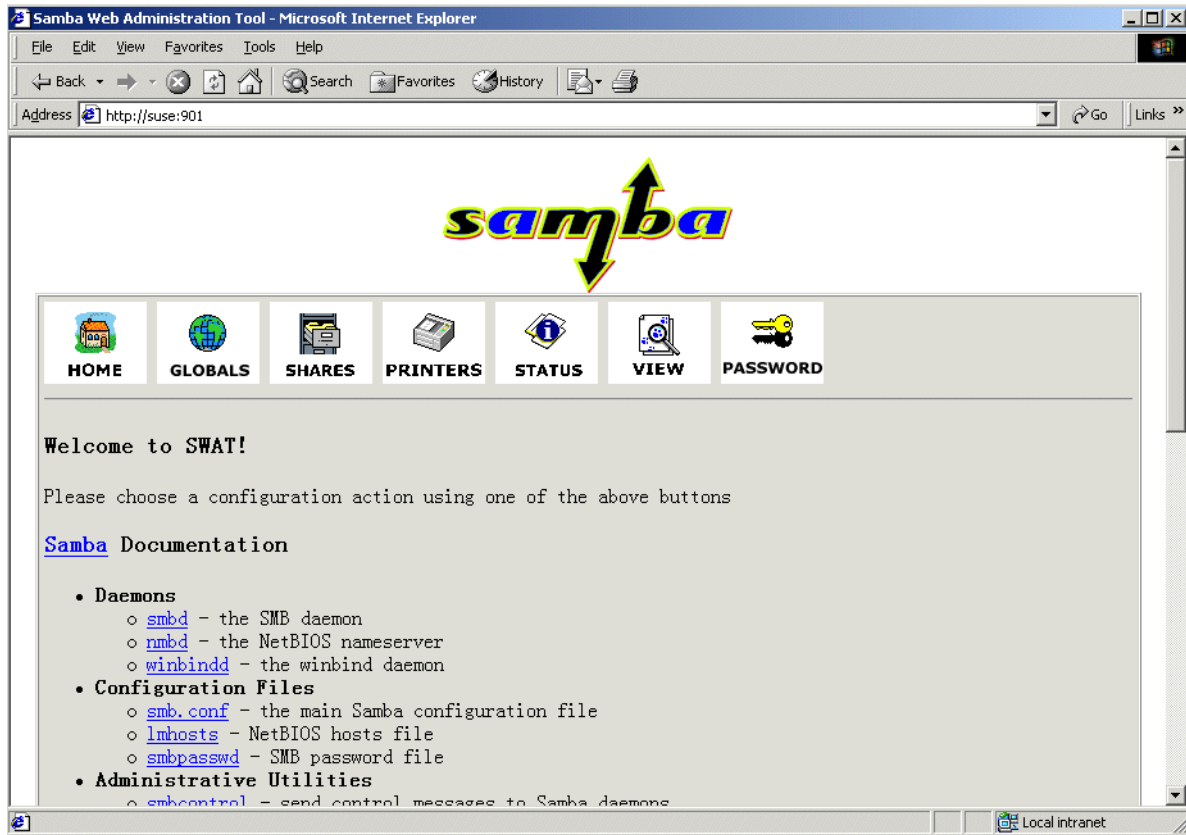


Figure 4-2 Connecting to SWAT

Then, click the **GLOBALS** icon and enter the relevant details for your server, similar to those we entered. We recommend that you use encrypted passwords as shown in Figure 4-3:

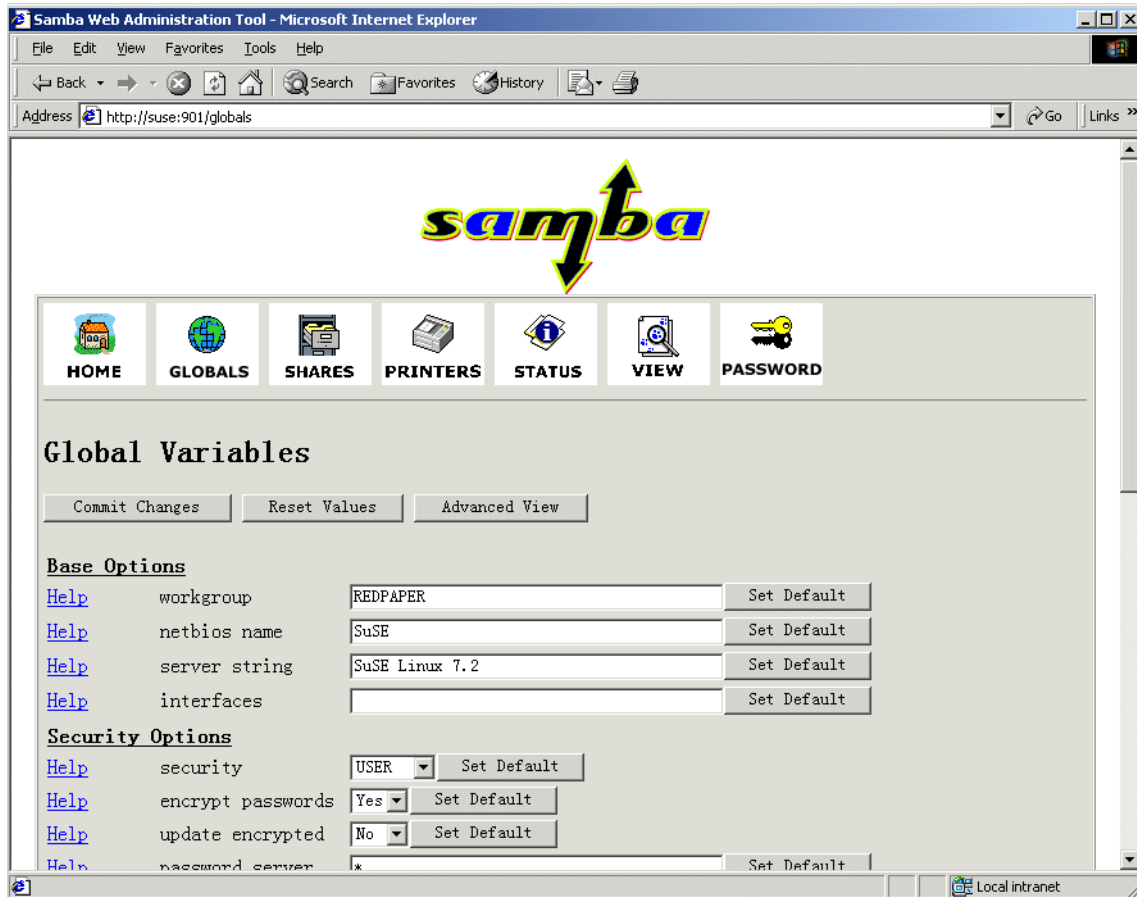


Figure 4-3 Setting up a simple Samba server

Finally, click **Commit Changes** to update smb.conf. After this, /etc/samba/smb.conf should look something like Example 4-4:

Example 4-4 A sample smb.conf file

```
# Global parameters
[global]
    workgroup = REDPAPER
    netbios name = SUSE
    server string = SUSE LINUX 7.2
    encrypt passwords = Yes
    log file = /var/log/samba/log.%m
    max log size = 50
    socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
    dns proxy = No

[homes]
    comment = Home Directories
    read only = No
    browseable = No

[printers]
    comment = All Printers
    path = /var/spool/samba
    printable = Yes
```

Before putting this file in production, we must make sure that it does not contain any errors. To assist you in this task, Samba provides a tool called `testparm` to check parameter values. Entering `testparm` at the command prompt tests `smb.conf` and reports any problems.

If everything checks out correctly, restart `nmbd` and `smbd` using SWAT.

4.5.6 Creating shares

Now that we have properly configured an SMB server, we can create some shares for the benefit of users. These are the steps to create a shared directory using SWAT:

1. Open the Web browser and connect to SWAT. Click **SHARES**.
2. Enter a name for the share you want to create in the Create Share field. Click **Create Share**.
3. Fill in the comment and path fields with the appropriate information.
4. Click **Commit Changes**.
5. Click **Status**.
6. Click **Restart smbd**.

Now you can test the share you have created. First test it on the Linux server. Run this command:

```
suse:~ # smbclient -L localhost
```

You will be asked for a password. Type the SMB password you set for the root account. If you have not changed it since you last used `mksmbpasswd.sh`, the password will be blank. You should see a list of shares available on the Linux server, including the share you created.

To allow access by your Windows users, create user accounts on the Linux server for each Windows user with the `useradd` command. Then use `smbpasswd` to set the matching Samba password. This password should be identical to the one the user has to enter to log in to Windows.

For example, if our Windows user is `winuser`, with a Windows password of `123`, Example 4-5 shows the commands to add the user account:

Example 4-5 Creating an account for a Windows user

```
suse:~ # useradd winuser
suse:~ # smbpasswd -a winuser
New SMB password:
Retype new SMB password:
Added user winuser.
```

It is now possible to access this share from Windows. If prompted for a password, the user should enter the one used to log in to Windows.

4.5.7 Working with Domain level security

To allow the Samba server to participate in a Windows 2000 domain as a member server, the steps below should be followed:

1. Stop `smbd` and `nmbd` on the Samba server.
2. Log on to the Windows 2000 domain.

3. Add a new computer account (the computer name should be identical to the NetBIOS name of the Samba server) in the Computers container of Active Directory Users and Computers, as shown in Figure 4-4:

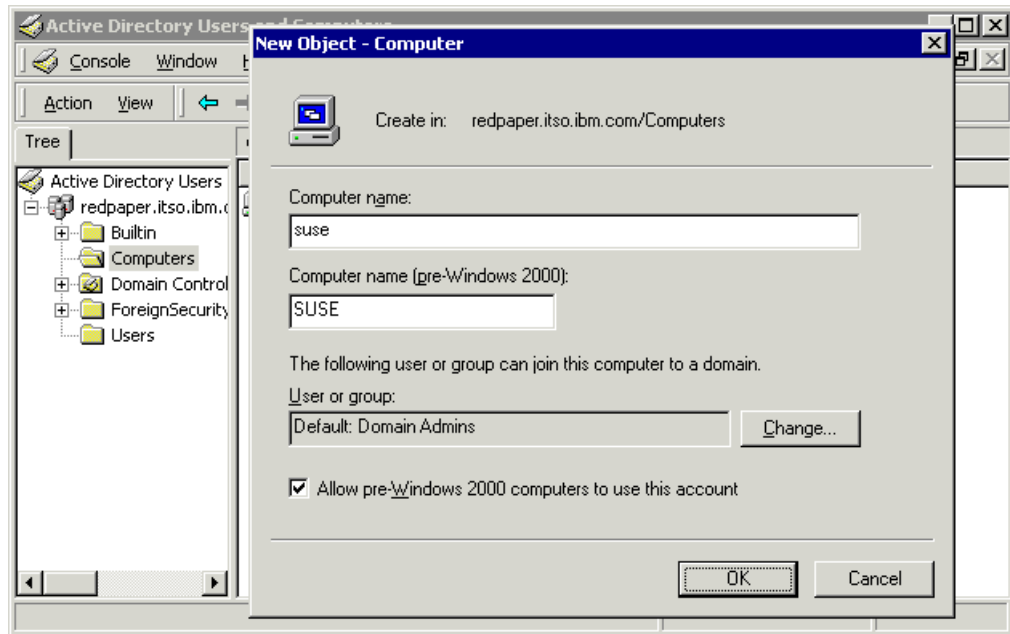


Figure 4-4 Creating a computer account for a Windows 2000 Domain

4. At the Linux server, enter the following command:

```
smbpasswd -j <NT_DOMAIN> -r <NT_PDC>,
```

where NT_DOMAIN is the Windows domain name, and NT_PDC is the computer name of the Domain Controller.

5. Access SWAT to modify the following parameters in the Globals section:

```
workgroup = NT_DOMAIN
password server = NT_PDC
security = domain
```

6. Start nmbd and smb.

Now, when Samba receives a service request, it passes the user name and password to the domain controller for authentication. If the user name and password are confirmed to represent a valid user in the domain, and the user has been configured to have access to the requested resource in smb.conf, access is granted. Otherwise access is denied.

Samba will try to access files on the Linux server as per the client request, using the permissions set to the user account on the Linux server (a Linux user account for each domain user must exist; otherwise the access will fail).

4.6 Setting up the SMB Server on Red Hat Linux 7.1

In this section, we build a simple Samba server to provide network clients with the requisite SMB services.

In the configuration we describe, we use the following settings:

- ▶ User level security

- ▶ Allow all hosts
- ▶ Encrypted passwords

The first step in configuring our Samba server is to make sure that the SMB service starts correctly during the boot process. Next, we configure SWAT for this distribution of Linux. Finally, we use SWAT to configure the Samba server to provide the network services for our clients.

4.6.1 Starting the SMB service during the boot process

As a system service, the SMB service should be configured to start during the boot process. In order to know how to modify system settings to start a service when Linux boots, let us take a look at its boot process.

When your server is powered on or restarted, the machine's power-on self test (POST) and basic input/output system (BIOS) check and initialize your system. Once this is complete, the server boots from the hard disk and, as for all other Linux distributions covered in this Redpaper, the Linux Boot Manager (LILO) is loaded.

LILO allows you to select bootup options if you wish, but if you do nothing Linux is selected and the kernel is automatically loaded and decompressed in the server's memory. The kernel then initializes device drivers and file systems, and calls a special program, named `init`.

It is from this point that things differ among Linux distributions.

Red Hat Linux 7.1 uses the System V (SysV) `init` program. In this model, all `init` scripts are located in `/etc/rc.d`. This model differs from BSD `init` in that the configuration files are held in a subdirectory of `/etc` instead of residing directly in `/etc`. The `rc.d` directory contains `rc.sysinit`, `rc.local` and directories such as `rc1.d`, `rc2.d`, and so on, which correspond to the different run levels we discussed in 2.2.1, "Linux daemons" on page 7.

When `init` is executed, it executes the following steps:

1. First, `rc.sysinit` is called.
2. Next, all scripts under the `rc?.d` directory (corresponding to the current run level) are executed.
3. Finally, `rc.local` is called, termination of which completes the initialization process.

Because `rc.sysinit` focuses on very basic system configurations such as bringing up network functions, it is not appropriate to place a command to start the SMB service in this file. You could add an SMB startup script in the `rc?.d` directory, but we recommend the simplest approach, which is to edit the `rc.local` file to start the SMB service.

The SMB service requires both the Network Message Block daemon (`nmbd`), which is the daemon responsible for NetBIOS name resolution, and the Server Message Block daemon (`smbd`), which is responsible for servicing the file and print service requests of SMB clients.

To start these daemons at boot time, edit `rc.local` to add the two following lines to the end of the file:

```
/usr/sbin/nmbd -D
/usr/sbin/smbd -D
```

Save the file and restart your server. When the server is running again, use the `ps` command to verify the result. The SMB service has started properly if you see the output as shown here:


```
[root@redhat /samba]# ps ax|grep smbd
1076 ?      S        0:00 smbd
```

The SMB service has started as a default system service and your server has now the basic ability to act as an SMB server. There is still some extra work that needs to be done. The next step will be to configure the Samba Web Admin Tool (SWAT).

4.6.2 Configuring SWAT on Red Hat Linux 7.1

Every time the SMB service starts, it opens its configuration file (`/etc/samba/smb.config`) to load the settings it contains. These settings cover all aspects of the SMB service. As we've discussed earlier, Samba comes with a Web-based configuration tool named SWAT, which is accessed using a Web browser through port 901. SWAT itself processes the Web requests (that is, it does not require Apache to run).

It is important to note that the SWAT process is not started when your Linux server boots up. Instead, a daemon such as `inetd` or `xinetd` have responsibility for starting it. With Red Hat Linux 7.1, `xinetd` performs that function and not `inetd`. Configuration files for the services controlled by the `xinetd` daemon are held in the directory `/etc/xinetd.d`.

The default configuration file for SWAT that comes with the Samba package is shown in Example 4-6:

Example 4-6 The default SWAT configuration file

```
# default: off
# description: SWAT is the Samba Web Admin Tool. Use swat \
#             to configure your Samba server. To use SWAT, \
#             connect to port 901 with your favorite web browser.
service swat
{
    port      = 901
    socket_type = stream
    wait      = no
    only_from = localhost
    user      = root
    server    = /usr/sbin/swat
    log_on_failure += USERID
    disable   = yes
}
```

As you can see, the SWAT service is disabled by default. To enable it, we need to modify this configuration file, which is itself called `swat`.

In Example 4-6, the lines between the curly brackets contain attribute-value pairs. Detailed information about the available attributes and their meanings can be found in the man pages for `xinetd.conf`. We used the settings shown in Example 4-7, which enable the service and allow access from the server and our management workstation (`ws`):

Example 4-7 Sample xinetd configuration file to enable SWAT

```
default: off
# description: SWAT is the Samba Web Admin Tool. Use swat \
#             to configure your Samba server. To use SWAT, \
#             connect to port 901 with your favorite web browser.
service swat
{
    port      = 901
    socket_type = stream
    wait      = no
    only_from = localhost ws #only allow access from localhost and computer ws
    user      = root
    server    = /usr/sbin/swat
    log_on_failure += USERID
    disable  = no
}
```

Once you have made the proper changes to the configuration file, restart the xinetd daemon in the following manner:

1. Get the xinetd daemon's Process ID (PID) by looking at the contents of the `/var/run/xinetd.pid`:

```
[root@redhat xinetd.d]# cat /var/run/xinetd.pid
680
```

In the example above, the PID is 680.

2. Kill and restart xinetd:

```
[root@redhat xinetd.d]# kill -TERM 680
[root@redhat xinetd.d]# xinetd
```

Now that this done, you can test your SWAT settings using the following method:

3. Use lynx (a text browser) on the Linux server.
 - a. Switch to a text-based terminal (Linux permits you to switch between terminals by pressing Alt+F1 through Alt+F6 for a text mode terminal or Alt+Ctrl+F1 through Alt+Ctrl+F6 for a graphic mode terminal).
 - b. Enter `lynx localhost:901` and wait for several seconds. You have successfully configured SWAT if you see the following message appear:
Username for 'SWAT' at server 'localhost:901':
 - c. Now you can switch to graphic mode and use a Web browser to access SWAT at:
<http://localhost:901>
4. If you want another workstation, specifically one running Windows, to be able to access the SWAT tool, you need to add the workstation's host name to the `only_from` attribute (as shown in Example 4-7).
 - Do not forget to restart xinetd.
 - If you do not have a DNS system that supports forward and reverse lookups for both computers, add the host names to the hosts file. The hosts file is located in the `/etc` directory on Red Hat Linux systems, `\windows` on Windows 98/ME systems and `\winnt\system32\drivers\etc\` on Windows NT/2000 systems.
 - Once you have done so, you can direct your browser to <http://sambaservername:901>. If you see a window requesting a user name and a password, then your configuration is successful. Otherwise, make sure you can access SWAT from the Linux server, then check the configuration file, restart xinetd and check the hosts file settings.

4.6.3 Migrating user accounts for Samba

Samba-specific accounts must be set up for your users before they can access the SMB services now provided by your Linux server. These accounts (and related information) are held in a file called `smbpasswd`.

Initially, the `smbpasswd` file does not exist and must be created using the `smbpasswd` command. When executed, the command will prompt you for the new Samba password and create a `smbpasswd` file.

It is also possible to create an encrypted `smbpasswd` file based on the `/etc/passwd` file by using the following command:

```
[root@redhat samba]# cat /etc/passwd | mksmbpasswd.sh > /etc/samba/smbpasswd
```

This ensures every person who has an account on the Linux server has a Samba account as well.

Note: Users who have been given a Samba account should change their own SMB password using the `smbpasswd` command.

4.6.4 Configuring Samba global settings

Global settings are a collection of attributes that define the behavior of Samba and the manner in which it delivers its services to SMB clients. Please consult 4.4.4, “Configuring global settings for Samba” on page 50 for a list of commonly used attributes.

4.6.5 Using SWAT to configure the Samba server

As was explained in 4.4.2, “Configuring the Samba Web Admin Tool (SWAT)” on page 49, Samba configuration is managed using a file called `/etc/samba/smb.conf`. You can modify this with a normal text editor or by using the Samba Web Admin Tool (SWAT), which, as its name implies, provides access through a Web browser. This is the tool we chose to use, as Windows users are generally more comfortable with a GUI interface. However, remember that using SWAT may change settings you may have made manually, so it is best to choose one method and use that exclusively.

Open your preferred Web browser and connect to SWAT as shown in Figure 4-5:

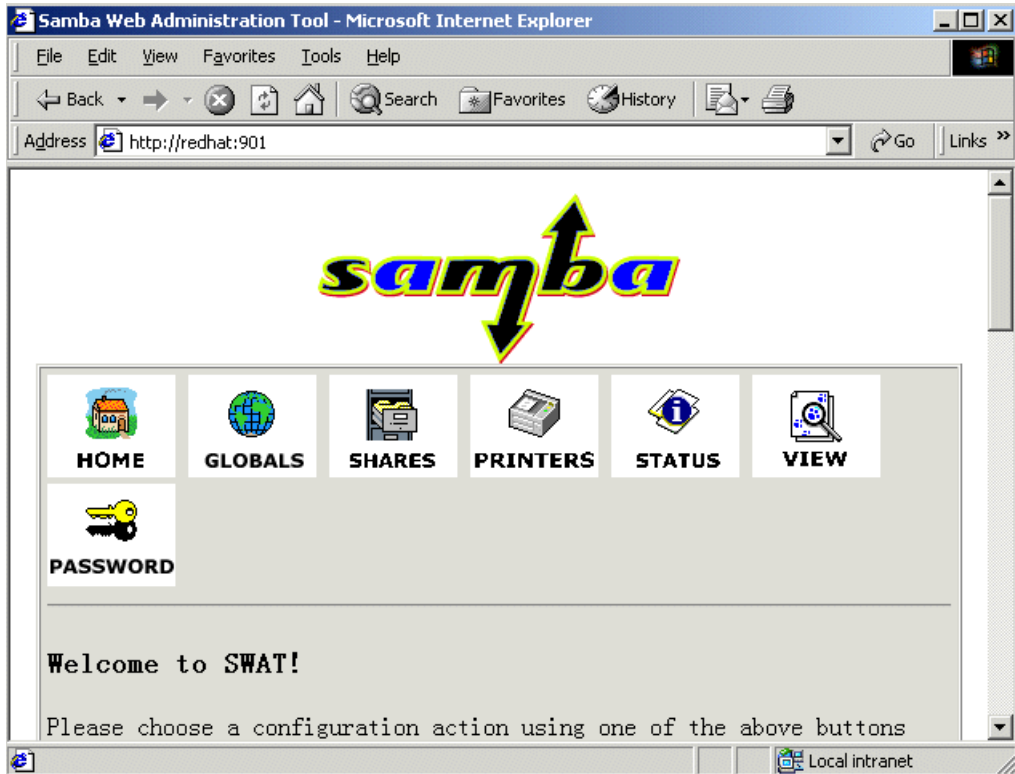


Figure 4-5 Connecting to SWAT

Then, click the **GLOBALS** icon and enter the relevant details for your server, similar to those we entered (shown in Figure 4-6):

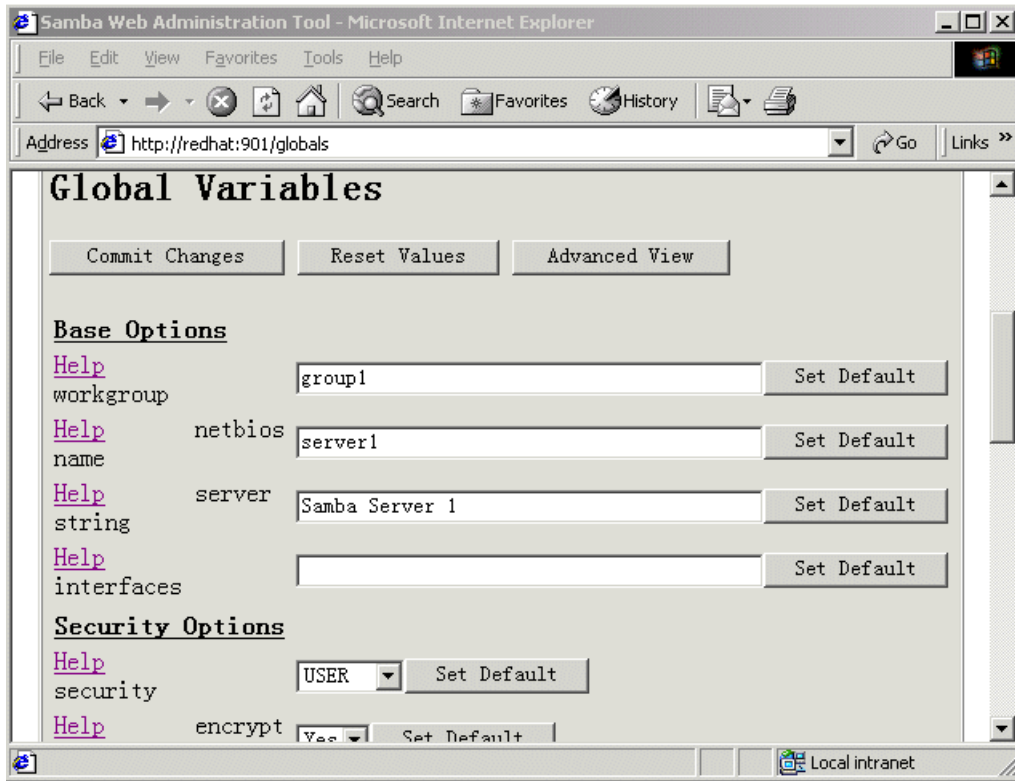


Figure 4-6 Setting up a simple Samba server

Note that we recommend that you use encrypted passwords.

Finally, click **Commit Changes** to update `smb.conf`. After this, `/etc/samba/smb.conf` should look something like Example 4-8:

Example 4-8 A sample `smb.conf` file

```
# Global parameters
[global]
    workgroup = GROUP1
    netbios name = SERVER1
    server string = Samba Server 1
    encrypt passwords = Yes
    log file = /var/log/samba/log.%m
    max log size = 50
    socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
    dns proxy = No

[homes]
    comment = Home Directories
    read only = No
    browseable = No

[printers]
    comment = All Printers
    path = /var/spool/samba
    printable = Yes
```

Before putting this file in production, we must make sure that it does not contain any errors. To assist you in this task, Samba provides a tool called `testparm` to check parameter values. Entering `testparm` at the command prompt tests `smb.conf` and reports any problems.

If everything checks out correctly, restart `nmbd` and `smbd` using SWAT.

4.6.6 Creating shares

Now that we have properly configured an SMB server, we can create some shares for the benefit of users. These are the steps to create a shared directory using SWAT:

1. Open the Web browser and connect to SWAT. Click **SHARES**.
2. Enter a name for the share you want to create in the Create Share field. Click **Create Share**.
3. Fill in the comment and path fields with the appropriate information.
4. Click **Commit Changes**.
5. Click **Status**.
6. Click **Restart smbd**.

Now you can test the share you have created. First test it on the Linux server. Run this command:

```
[root@redhat samba]# smbclient -L localhost
```

You will be asked for a password. Type the SMB password you set for the root account. If you have not changed it since you last used `mksmbpasswd.sh`, the password will be blank. You should see a list of shares available on the Linux server, including the share you created.

To allow access by your Windows users, create user accounts on the Linux server for each Windows user with the `useradd` command. Then use `smbpasswd` to set the matching Samba password. This password should be identical to the one the user has to enter to log in to Windows.

For example, if our Windows user is `winuser`, with a Windows password of `123`, Example 4-5 shows the commands to add the user account:

Example 4-9 Creating an account for a Windows user

```
[root@redhat /root]# useradd winuser
[root@redhat /root]# smbpasswd -a winuser
New SMB password:
Retype new SMB password:
Added user winuser.
```

It is now possible to access this share from Windows. If prompted for a password, the user should enter the one used to log in to Windows.

4.6.7 Working with domain level security

To allow the Samba server to participate in a Windows 2000 domain as a member server, the steps below should be followed:

1. Stop `smbd` and `nmbd` on the Samba server.
2. Log on to the Windows 2000 domain.

3. Add a new computer account (the computer name should be identical to the NetBIOS name of the Samba server) in the Computers container of Active Directory Users and Computers, as shown in Figure 4-7:

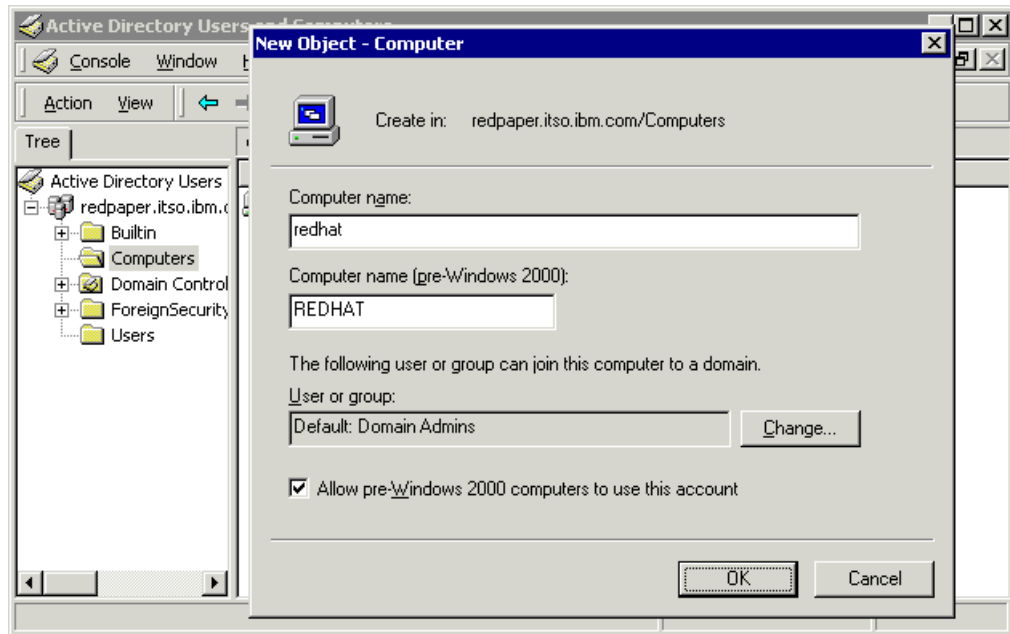


Figure 4-7 Creating a computer account for the Windows 2000 domain

4. At the Linux server, enter the following command:

```
smbpasswd -j <NT_DOMAIN> -r <NT_PDC>,
```

where NT_DOMAIN is the Windows domain name, and NT_PDC is the computer name of the Domain Controller.

5. Access SWAT to modify the following parameters in the Globals section:

```
workgroup = NT_DOMAIN
password server = NT_PDC
security = domain
```

6. Start nmbd and smb.

Now, when Samba receives a service request, it passes the user name and password to the domain controller for authentication. If the user name and password are confirmed to represent a valid user in the domain, and the user has been configured to have access to the requested resource in smb.conf, access is granted. Otherwise access is denied.

Samba will try to access files on the Linux server as per the client request, using the permissions set to the user account on the Linux server (a Linux user account for each domain user must exist; otherwise the access will fail).

4.7 Setting up the SMB Server on Caldera OpenLinux 3.1

In this section, we build a simple Samba server to provide network clients with the requisite SMB services.

In the configuration we describe, we use the following settings:

- ▶ User level security

- ▶ Allow all hosts
- ▶ Encrypted passwords

The first step in configuring our Samba server is to make sure that the SMB service starts correctly during the boot process. Next, we configure SWAT for this distribution of Linux. Finally, we use SWAT to configure the Samba server to provide the network services for our clients.

4.7.1 Starting the SMB service during the boot process

As a system service, the SMB service should be configured to start during the boot process. In order to know how to modify system settings to start a service when Linux boots, let us take a look at its boot process.

When your server is powered on or restarted, the machine's power-on self test (POST) and basic input/output system (BIOS) check and initialize your system. Once this is complete, the server boots from the hard disk and, as for all other Linux distributions covered in this Redpaper, the Linux Boot Manager (LILO) is loaded.

LILO allows you to select bootup options if you wish, but if you do nothing Linux is selected and the kernel is automatically loaded and decompressed in the server's memory. The kernel then initializes device drivers and file systems, and calls a special program, named `init`.

It is from this point that things differ among Linux distributions.

Caldera OpenLinux 3.1 uses the System V (SysV) `init` program. In this model, all `init` scripts are located in `/etc/rc.d`. This model differs from BSD `init` in that the configuration files are held in a subdirectory of `/etc` instead of residing directly in `/etc`. The `rc.d` directory contains `rc.sysinit`, `rc.local` and directories such as `rc1.d`, `rc2.d`, and so on, which correspond to the different run levels we discussed in 2.2.1, "Linux daemons" on page 7.

When `init` is executed, it executes the following steps:

1. First, `rc.sysinit` is called.
2. Next, all scripts under the `rc?.d` directory (corresponding to the current run level) are executed.
3. Finally, `rc.local` is called, termination of which completes the initialization process.

Because `rc.sysinit` focuses on very basic system configurations such as bringing up network functions, it is not appropriate to place a command to start the SMB service in this file. You could add an SMB startup script in the `rc?.d` directory, but we recommend the simplest approach, which is to edit the `rc.local` file to start the SMB service.

The SMB service requires both the Network Message Block daemon (`nmbd`), which is the daemon responsible for NetBIOS name resolution, and the Server Message Block daemon (`smbd`), which is responsible for servicing the file and print service requests of SMB clients.

To start these daemons at boot time, edit `rc.local` to add the two following lines to the end of the file:

```
/usr/sbin/nmbd -D
/usr/sbin/smbd -D
```

Save the file and restart your server. When the server is running again, use the `ps` command to verify the result. The SMB service has started properly if you see the output as shown here:


```
[root@redhat /samba]# ps ax|grep smbd
1076 ?      S        0:00 smbd
```

The SMB service has started as a default system service and your server has now the basic ability to act as an SMB server. There is still some extra work that needs to be done. The next step will be to configure the Samba Web Admin Tool (SWAT).

4.7.2 Configuring SWAT on Caldera OpenLinux 3.1

Every time the SMB service starts, it opens its configuration file (`/etc/samba/smb.config`) to load the settings it contains. These settings cover all aspects of the SMB service. As we have discussed earlier, Samba comes with a Web-based configuration tool named SWAT, which is accessed using a Web browser through port 901. SWAT itself processes the Web requests (that is, it does not require Apache to run).

It is important to note that the SWAT process is not started when your Linux server boots up. Instead, a daemon such as `inetd` or `xinetd` has the responsibility for starting it. With Caldera OpenLinux 3.1, `inetd` performs that function. `/etc/inetd.conf` is the default configuration file for the `inetd` daemon. This file enables you to specify the daemons to start by default and to supply the arguments that correspond to the required mode of operation for each daemon.

Each line in this file is of the form:

```
ServiceName SocketType ProtocolName Wait/NoWait UserName ServerPath ServerArgs
```

These fields must be delimited by spaces or tabs, and their meaning is as follows:

- ▶ `ServiceName` contains the name of an Internet service defined in the `/etc/services` file.
- ▶ `SocketType` contains the name for the type of socket used for the service. Possible values for the `SocketType` parameter are:
 - `stream` specifies that a stream socket is used for the service.
 - `dgram` specifies that a datagram socket is used for the service.
 - `sunrpc_tcp` specifies that a Sun remote procedure call (RPC) socket is used for the service, over a stream connection.
 - `sunrpc_udp` specifies that a Sun RPC socket is used for the service, over a datagram connection.
- ▶ `ProtocolName` contains the name of an Internet protocol defined in the `/etc/protocols` file. For example, use the `tcp` value for a service that uses TCP/IP and the `udp` value for a service that uses the User Datagram Protocol (UDP).
- ▶ `Wait/NoWait` contains either the `wait` or the `nowait` instruction for datagram sockets and the `nowait` instruction for stream sockets. The `Wait/NoWait` field determines whether the `inetd` daemon waits for a datagram server to release the socket before continuing to listen at the socket.
- ▶ `UserName` specifies the user name that the `inetd` daemon should use to start the server. This variable allows a server to be given less permission than the root user.
- ▶ `ServerPath` specifies the full path name of the server that the `inetd` daemon should execute to provide the service.
- ▶ `ServerArgs` specifies the command line arguments that the `inetd` daemon should use to execute the server. The maximum number of arguments is five. The first argument specifies the name of the server used. If the `SocketType` parameter is `sunrpc_tcp` or `sunrpc_udp`, the second argument specifies the program name and the third argument specifies the release of the program. For services that the `inetd` daemon provides internally, this field should be empty.

To configure SWAT, follow these steps:

1. Edit `/etc/inetd.conf` to include this line:

```
swat stream tcp    nowait.400    root    /usr/sbin/swat  swat
```

and save the file.

2. Terminate and restart `inetd`.

Now that this done, you can test your SWAT settings using the following method:

3. Using `lynx` (a text browser) on the Linux server.

- a. Switch to a text-based terminal (Linux permits you to switch between terminals by pressing `Alt+F1` through `Alt+F6` for a text mode terminal or `Alt+Ctrl+F1` through `Alt+Ctrl+F6` for a graphic mode terminal).

- b. Enter `lynx localhost:901` and wait for several seconds. You have successfully configured SWAT if you see the following message appear:

```
Username for 'SWAT' at server 'localhost:901':
```

- c. Now you can switch to graphic mode and use a Web browser to access SWAT at:

```
http://localhost:901
```

4. If you want another workstation, specifically one running Windows, to be able to access the SWAT tool, you need edit the entry for SWAT in the file `/etc/hosts.deny`:

```
swat:ALL EXCEPT 127.0.0.1 w2k-300
```

In this example, we are allowing `localhost` and `w2k-300` to access SWAT.

- If you do not have a DNS system that supports forward and reverse lookups for both computers, add the host names to the `hosts` file. The `hosts` file is located in the `/etc` directory on Caldera OpenLinux systems, `\windows` on Windows 98/Me systems and `\winnt\system32\drivers\etc\` on Windows NT/2000 systems.

Once you have done so, you can direct your browser to <http://sambaservername:901>. If you see a window requesting a user name and a password, then your configuration is successful. Otherwise, make sure you can access SWAT from the Linux server first, then check the configuration file.

4.7.3 Migrating user accounts for Samba

Samba-specific accounts must be set up for your users before they can access the SMB services now provided by your Linux server. These accounts (and related information) are held in a file called `smbpasswd`.

Initially, the `smbpasswd` file does not exist and must be created using the `smbpasswd` command. When executed, the command will prompt you for the new Samba password and create a `smbpasswd` file.

It is also possible to create an encrypted `smbpasswd` file based on the `/etc/passwd` file by using the following command:

```
[root@caldera /root]# cat /etc/passwd | mksmbpasswd.sh > /etc/samba/smbpasswd
```

This ensures every person who has an account on the Linux server has a Samba account as well.

Note: Users who have been given a Samba account should change their own SMB password using the `smbpasswd` command.

4.7.4 Configuring Samba global settings

Global settings are a collection of attributes that define the behavior of Samba and the manner in which it delivers its services to SMB clients. Please consult 4.4.4, “Configuring global settings for Samba” on page 50 for a list of commonly used attributes.

4.7.5 Using SWAT to configure the Samba server

As was explained in 4.4.2, “Configuring the Samba Web Admin Tool (SWAT)” on page 49, Samba configuration is managed using a file called `/etc/samba/smb.conf`. You can modify this with a normal text editor or by using the Samba Web Admin Tool (SWAT), which, as its name implies, provides access through a Web browser. This is the tool we chose to use as Windows users are generally more comfortable with a GUI interface. However, remember that using SWAT may change settings you may have made manually, so it is best to choose one method and use that exclusively.

Open your preferred Web browser and connect to SWAT as shown in Figure 4-8:

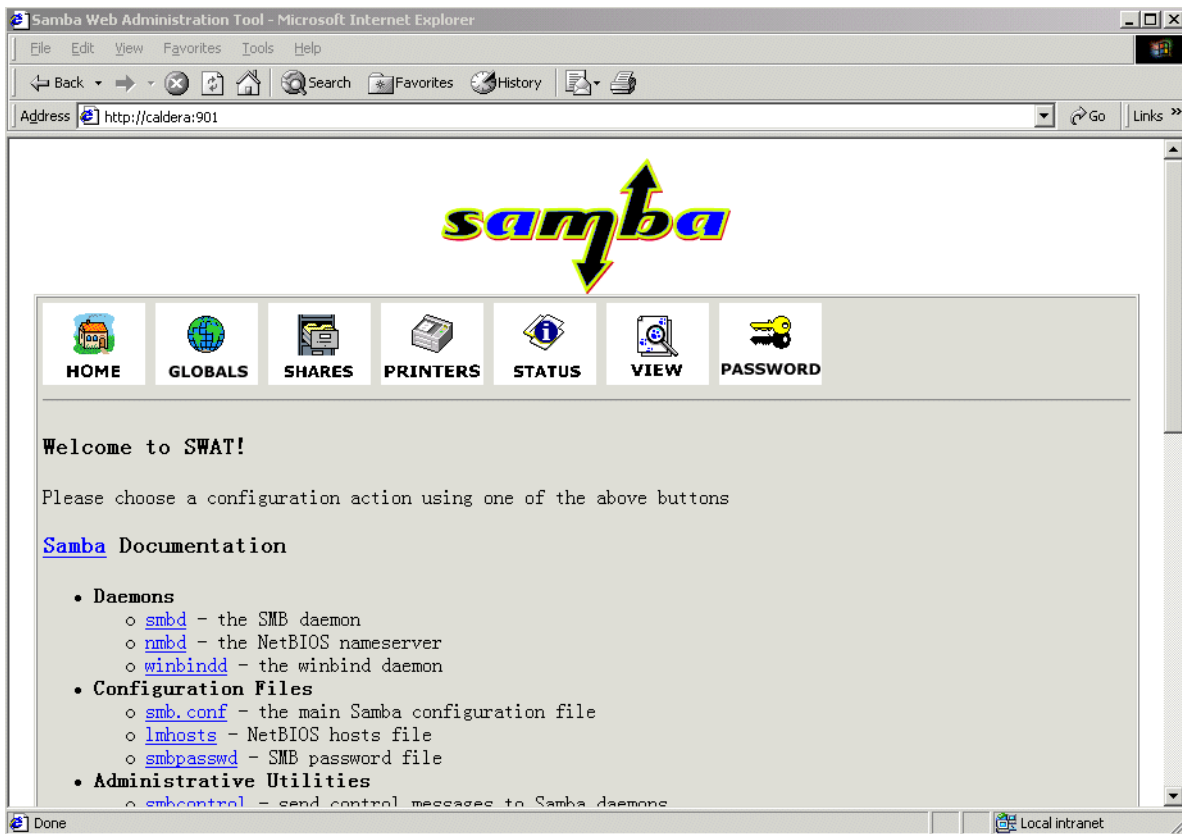


Figure 4-8 Connecting to SWAT

Then, click the **GLOBALS** icon and enter the relevant details for your server, similar to those we entered. We recommend that you use encrypted passwords as shown in Figure 4-9:

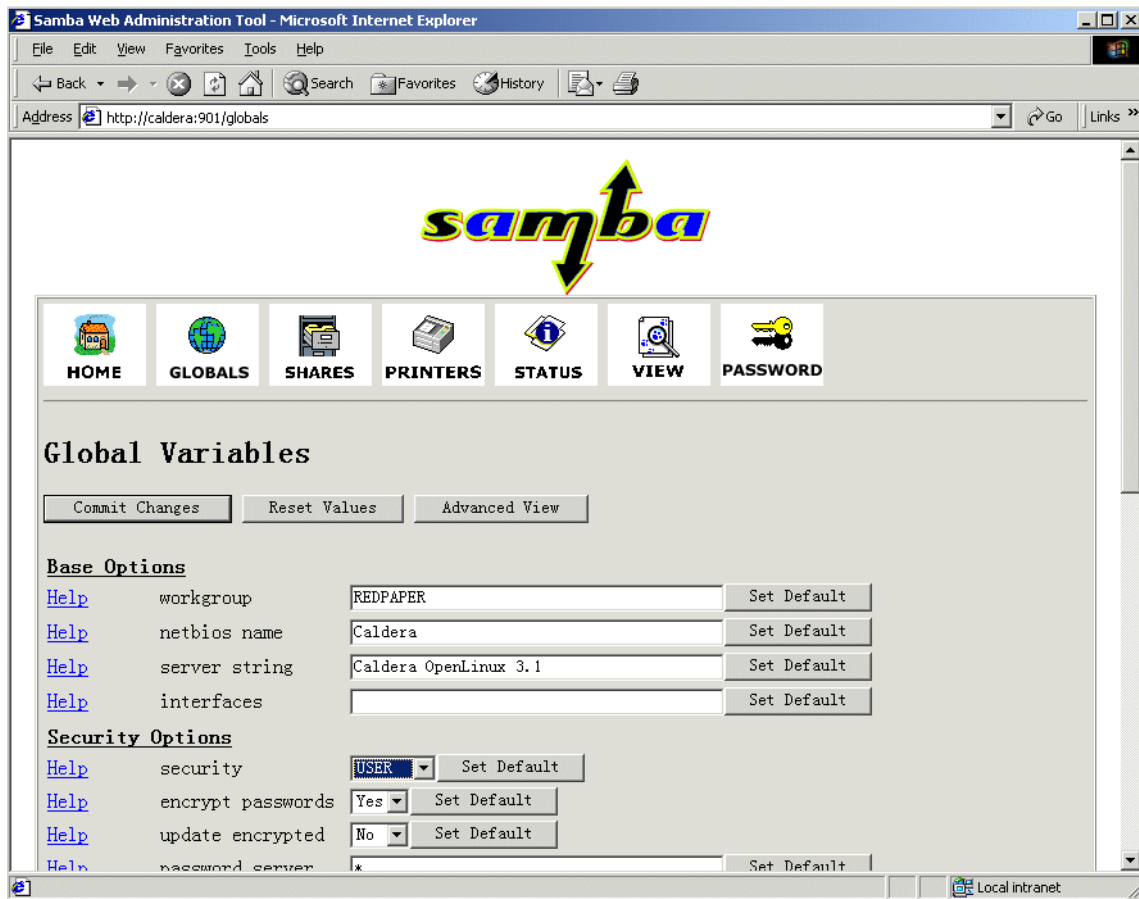


Figure 4-9 Setting up a simple Samba server

Finally, click **Commit Changes** to update smb.conf. After this, /etc/samba/smb.conf should look something like Example 4-10:

Example 4-10 A sample smb.conf file

```
# Global parameters
[global]
    workgroup = REDPAPER
    netbios name = CALDERA
    server string = CALDERA OPENLINUX 3.1
    security = USER
    encrypt passwords = Yes

[homes]
    comment = Home Directories
    read only = No
    browseable = No

[printers]
    comment = All Printers
    path = /var/spool/samba
    printable = Yes
```

Before putting this file in production, we must make sure that it does not contain any errors. To assist you in this task, Samba provides a tool called `testparm` to check parameter values. Entering `testparm` at the command prompt tests `smb.conf` and reports any problems.

If everything checks out correctly, restart `nmbd` and `smbd` using SWAT.

4.7.6 Creating shares

Now that we have properly configured an SMB server, we can create some shares for the benefit of users. These are the steps to create a shared directory using SWAT:

1. Open the Web browser and connect to SWAT. Click **SHARES**.
2. Enter a name for the share you want to create in the Create Share field. Click **Create Share**.
3. Fill in the comment and path fields with the appropriate information.
4. Click **Commit Changes**.
5. Click **Status**.
6. Click **Restart smbd**.

Now you can test the share you have created. First test it on the Linux server. Run this command:

```
[root@caldera samba]# smbclient -L localhost
```

You will be asked for a password. Type the SMB password you set for the root account. If you have not changed it since you last used `mksmbpasswd.sh`, the password will be blank. You should see a list of shares available on the Linux server, including the share you created.

To allow access by your Windows users, create user accounts on the Linux server for each Windows user with the `useradd` command. Then use `smbpasswd` to set the matching Samba password. This password should be identical to the one the user has to enter to log in to Windows.

For example, if our Windows user is `winuser`, with a Windows password of `123`, Example 4-5 shows the commands to add the user account:

Example 4-11 Creating an account for a Windows user

```
[root@caldera /root]# useradd winuser
[root@caldera /root]# smbpasswd -a winuser
New SMB password:
Retype new SMB password:
Added user winuser.
```

It is now possible to access this share from Windows. If prompted for a password, the user should enter the one used to log in to Windows.

4.7.7 Working with Domain level security

To allow the Samba server to participate in a Windows 2000 domain as a member server, the steps below should be followed:

1. Stop `smbd` and `nmbd` on the Samba server.
2. Log on to the Windows 2000 domain.

3. Add a new computer account (the computer name should be identical to the NetBIOS name of the Samba server) in the Computers container of Active Directory Users and Computers, as shown in Figure 4-10:

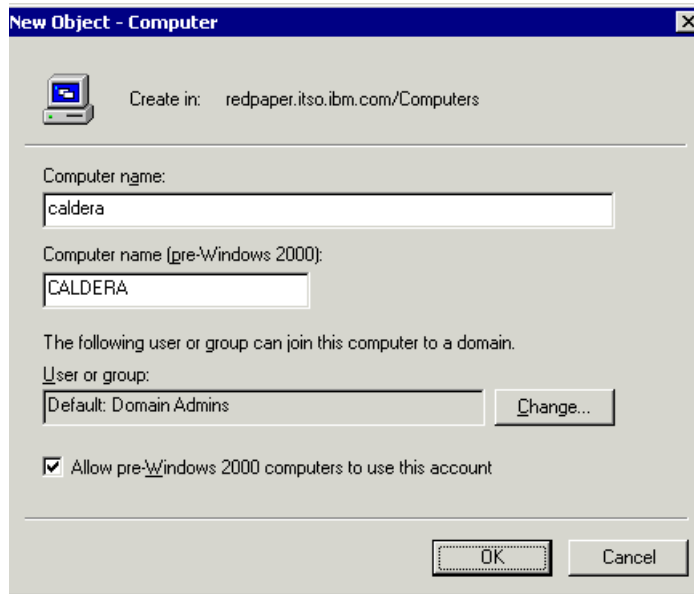


Figure 4-10 Creating a computer account for the Windows 2000 Domain

4. At the Linux server, enter the following command:

```
smbpasswd -j <NT_DOMAIN> -r <NT_PDC> ,
```

where NT_DOMAIN is the Windows domain name, and NT_PDC is the computer name of the domain controller.

5. Access SWAT to modify the following parameters in the Globals section:

```
workgroup = NT_DOMAIN  
password server = NT_PDC  
security = domain
```

6. Start nmbd and smbdc.

Now, when Samba receives a service request, it passes the user name and password to the domain controller for authentication. If the user name and password are confirmed to represent a valid user in the domain, and the user has been configured to have access to the requested resource in smb.conf, access is granted. Otherwise access is denied.

Samba will try to access files on the Linux server as per the client request, using the permissions set to the user account on the Linux server (a Linux user account for each domain user must exist; otherwise the access will fail).

4.8 Setting up the SMB Server on TurboLinux Server 6.5

In this section, we build a simple Samba server to provide network clients with the requisite SMB services.

In the configuration we describe, we use the following settings:

- ▶ User level security

- ▶ Allow all hosts
- ▶ Encrypted passwords

The first step in configuring our Samba server is to make sure that the SMB service starts correctly during the boot process. Next, we configure SWAT for this distribution of Linux. Finally, we use SWAT to configure the Samba server to provide the network services for our clients.

4.8.1 Starting the SMB service during the boot process

As a system service, the SMB service should be configured to start during the boot process. In order to know how to modify system settings to start a service when Linux boots, let us take a look at its boot process.

When your server is powered on or restarted, the machine's power-on self test (POST) and basic input/output system (BIOS) check and initialize your system. Once this is complete, the server boots from the hard disk and, as for all other Linux distributions covered in this Redpaper, the Linux Boot Manager (LILO) is loaded.

LILO allows you to select bootup options if you wish, but if you do nothing Linux is selected and the kernel is automatically loaded and decompressed in the server's memory. The kernel then initializes device drivers and file systems, and calls a special program, named **init**.

It is from this point that things differ among Linux distributions.

TurboLinux uses the System V (SysV) **init** program. In this model, all init scripts are located in `/etc/rc.d`. This model differs from BSD init in that the configuration files are held in a subdirectory of `/etc` instead of residing directly in `/etc`. The `rc.d` directory contains `rc.sysinit`, `rc.local` and directories such as `rc1.d`, `rc2.d`, and so on, which correspond to the different run levels we discussed in 2.2.1, "Linux daemons" on page 7.

When **init** is executed, it executes the following steps:

1. First, `rc.sysinit` is called.
2. Next, all scripts under the `rc?.d` directory (corresponding to the current run level) are executed.
3. Finally, `rc.local` is called, termination of which completes the initialization process.

Because `rc.sysinit` focuses on very basic system configurations such as bringing up network functions, it is not appropriate to place a command to start the SMB service in this file. You could add an SMB startup script in the `rc?.d` directory, but we recommend the simplest approach, which is to edit the `rc.local` file to start the SMB service.

The SMB service requires both the Network Message Block daemon (`nmbd`), which is the daemon responsible for NetBIOS name resolution, and the Server Message Block daemon (`smbd`), which is responsible for servicing the file and print service requests of SMB clients.

To start these daemons at boot time, edit `rc.local` to add the two following lines to the end of the file:

```
/usr/local/samba/bin/nmbd -D
/usr/local/samba/bin/smbd -D
```

Save the file and restart your server. When the server is running again, use the `ps` command to verify the result. The SMB service has started properly if you see the output as shown here:

```
[root@redhat /samba]# ps ax|grep smbd
1076 ?        S          0:00 smbd
```

The SMB service has started as a default system service and your server has now the basic ability to act as an SMB server. There is still some extra work that needs to be done. The next step will be to configure the Samba Web Admin Tool (SWAT).

4.8.2 Configuring SWAT on TurboLinux 6.5

Every time the SMB service starts, it opens its configuration file (`/usr/local/samba/lib`) to load the settings it contains. These settings cover all aspects of the SMB service. As we have discussed earlier, Samba comes with a Web-based configuration tool named SWAT, which is accessed using a Web browser through port 901. SWAT itself processes the Web requests (that is, it does not require Apache to run).

It is important to note that the SWAT process is not started when your Linux server boots up. Instead, a daemon such as `inetd` or `xinetd` has the responsibility for starting it. With TurboLinux 6.5, `inetd` performs that function. `/etc/inetd.conf` is the default configuration file for the `inetd` daemon. This file enables you to specify the daemons to start by default and to supply the arguments that correspond to the required mode of operation for each daemon.

Each line in this file is of the form:

```
ServiceName SocketType ProtocolName Wait/NoWait UserName ServerPath ServerArgs
```

These fields must be delimited by spaces or tabs, and their meaning is as follows:

- ▶ `ServiceName` contains the name of an Internet service defined in the `/etc/services` file.
- ▶ `SocketType` contains the name for the type of socket used for the service. Possible values for the `SocketType` parameter are:
 - `stream` specifies that a stream socket is used for the service.
 - `dgram` specifies that a datagram socket is used for the service.
 - `sunrpc_tcp` specifies that a Sun remote procedure call (RPC) socket is used for the service, over a stream connection.
 - `sunrpc_udp` specifies that a Sun RPC socket is used for the service, over a datagram connection.
- ▶ `ProtocolName` contains the name of an Internet protocol defined in the `/etc/protocols` file. For example, use the `tcp` value for a service that uses TCP/IP and the `udp` value for a service that uses the User Datagram Protocol (UDP).
- ▶ `Wait/NoWait` contains either the `wait` or the `nowait` instruction for datagram sockets and the `nowait` instruction for stream sockets. The `Wait/NoWait` field determines whether the `inetd` daemon waits for a datagram server to release the socket before continuing to listen at the socket.
- ▶ `UserName` specifies the user name that the `inetd` daemon should use to start the server. This variable allows a server to be given less permission than the root user.
- ▶ `ServerPath` specifies the full path name of the server that the `inetd` daemon should execute to provide the service.
- ▶ `ServerArgs` specifies the command line arguments that the `inetd` daemon should use to execute the server. The maximum number of arguments is five. The first argument specifies the name of the server used. If the `SocketType` parameter is `sunrpc_tcp` or `sunrpc_udp`, the second argument specifies the program name and the third argument specifies the release of the program. For services that the `inetd` daemon provides internally, this field should be empty.

To configure SWAT, follow these steps:

1. Edit `/etc/inetd.conf` to include this line:

```
swat stream tcp    nowait.400    root    /usr/sbin/swat  swat
```

and save the file.

2. Terminate and restart `inetd`.

Now that this done, you can test your SWAT settings using the following method:

3. Using `lynx` (a text browser) on the Linux server.

- a. Switch to a text-based terminal (Linux permits you to switch between terminals by pressing `Alt+F1` through `Alt+F6` for a text mode terminal, or `Alt+Ctrl+F1` through `Alt+Ctrl+F6` for a graphic mode terminal).

- b. Enter `lynx localhost:901` and wait for several seconds. You have successfully configured SWAT if you see the following message appear:

```
Username for 'SWAT' at server 'localhost:901':
```

- c. Now you can switch to graphic mode and use a Web browser to access SWAT at:

```
http://localhost:901
```

4. If you want another workstation, specifically one running Windows, to be able to access the SWAT tool, you need edit the entry for SWAT in the file `/etc/hosts.deny`:

```
swat:ALL EXCEPT 127.0.0.1 w2k-300
```

In this example, we are allowing `localhost` and `w2k-300` to access SWAT.

- If you do not have a DNS system that supports forward and reverse lookups for both computers, add the host names to the `hosts` file. The `hosts` file is located in the `/etc` directory on TurboLinux V6.5 systems, `\windows` on Windows 98/Me systems and `\winnt\system32\drivers\etc\` on Windows NT/2000 systems.

Once you have done so, you can direct your browser to <http://sambaservername:901>. If you see a window requesting a user name and a password, then your configuration is successful. Otherwise, make sure you can access SWAT from the Linux server first, then check the configuration file.

4.8.3 Migrating user accounts for Samba

Samba-specific accounts must be set up for your users before they can access the SMB services now provided by your Linux server. These accounts (and related information) are held in a file called `smbpasswd`.

Initially, the `smbpasswd` file does not exist and must be created using the `smbpasswd` command. When executed, the command will prompt you for the new Samba password and create a `smbpasswd` file.

It is also possible to create an encrypted `smbpasswd` file based on the `/etc/passwd` file by using the following command:

```
[root@turbo /root]# cat /etc/passwd | mksmbpasswd.sh > /etc/samba/smbpasswd
```

This ensures every person who has an account on the Linux server has a Samba account as well.

Note: Users who have been given a Samba account should change their own SMB password using the `smbpasswd` command.

4.8.4 Configuring Samba global settings

Global settings are a collection of attributes that define the behavior of Samba and the manner in which it delivers its services to SMB clients. Please consult 4.4.4, “Configuring global settings for Samba” on page 50 for a list of commonly used attributes.

4.8.5 Using SWAT to configure the Samba server

As was explained in 4.4.2, “Configuring the Samba Web Admin Tool (SWAT)” on page 49, Samba configuration is managed using a file called `/etc/samba/smb.conf`. You can modify this with a normal text editor or by using the Samba Web Admin Tool (SWAT), which, as its name implies, provides access through a Web browser. This is the tool we chose to use as Windows users are generally more comfortable with a GUI interface. However, remember that using SWAT may change settings you may have made manually, so it is best to choose one method and use that exclusively.

Open your preferred Web browser and connect to SWAT as shown in Figure 4-11:

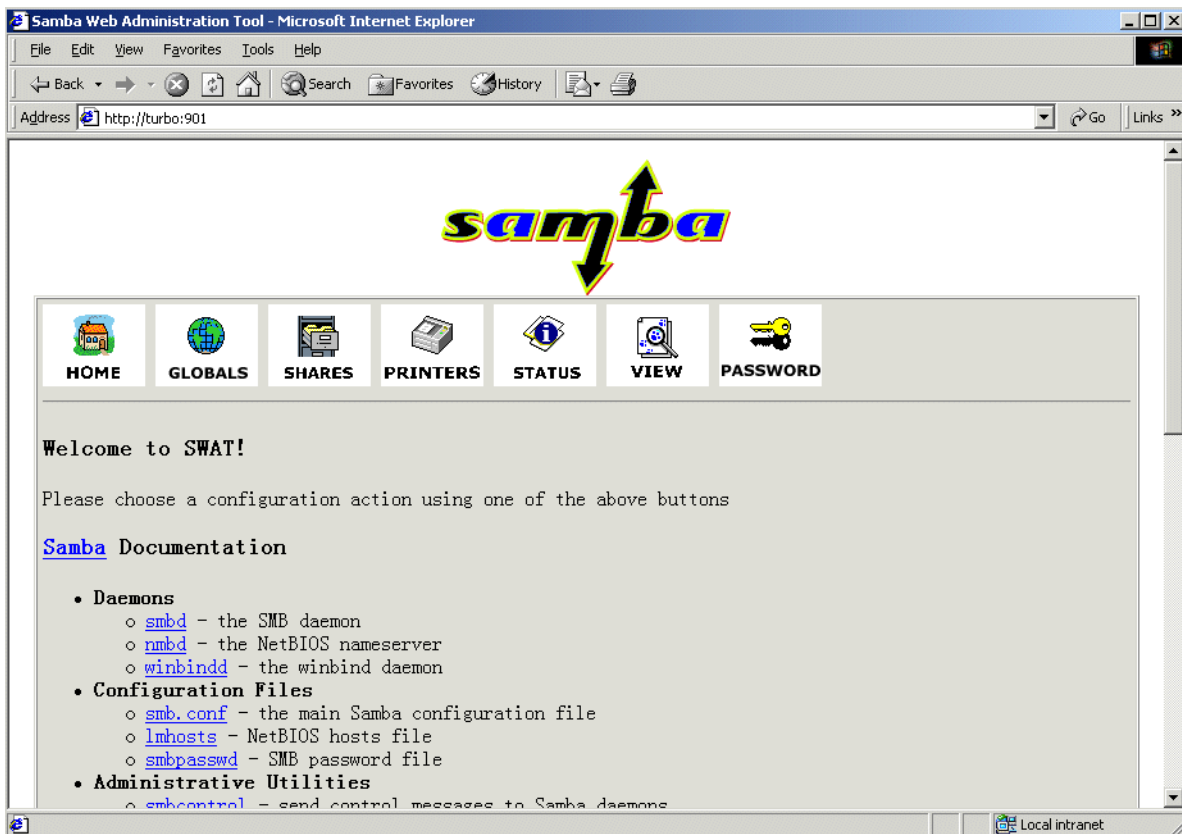


Figure 4-11 Connecting to SWAT

Then, click the **GLOBALS** icon and enter the relevant details for your server, similar to those we entered. We recommend that you use encrypted passwords as shown in Figure 4-12:

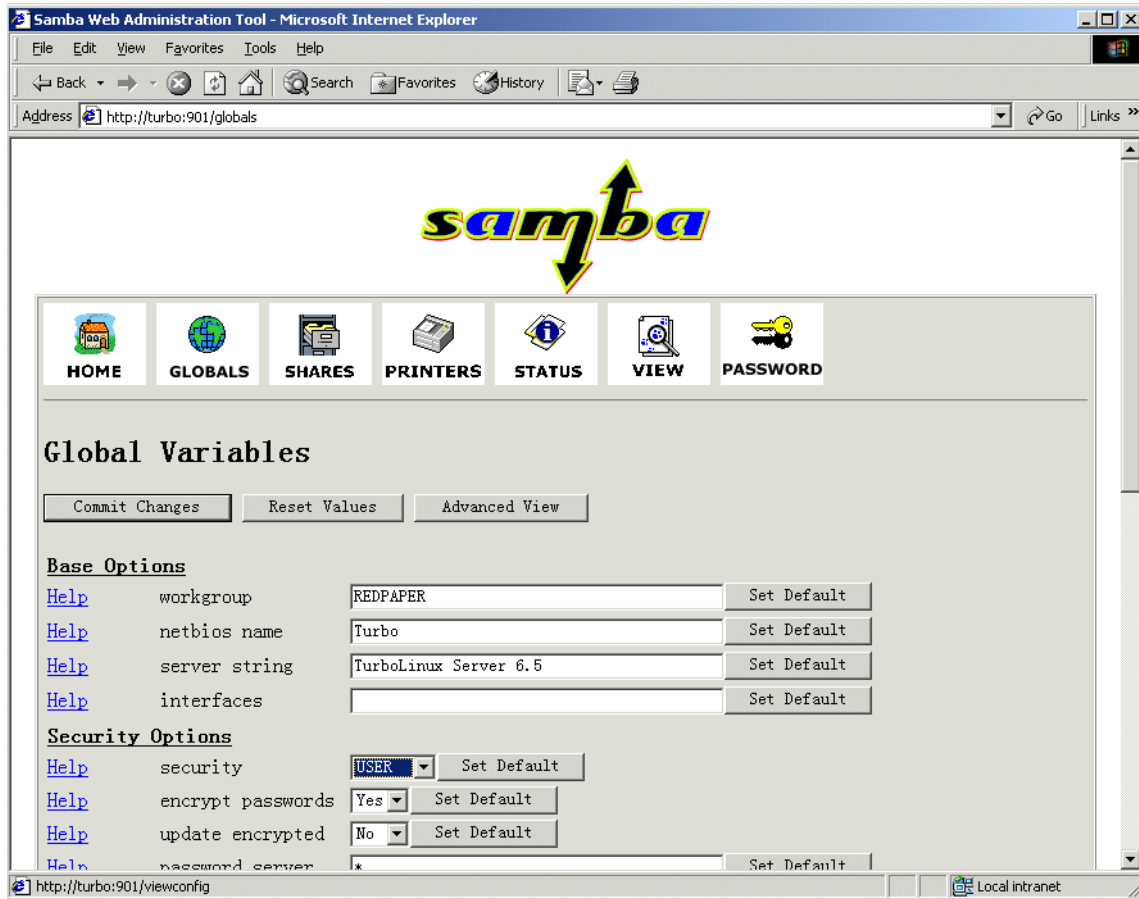


Figure 4-12 Setting up a simple Samba server

Finally, click **Commit Changes** to update smb.conf. After this, /etc/samba/smb.conf should look something like Example 4-12:

Example 4-12 A sample smb.conf file

```
# Global parameters
[global]
    workgroup = REDPAPER
    netbios name = TURBO
    server string = TURBOLINUX SERVER 6.5
    security = USER
    encrypt passwords = Yes

[homes]
    comment = Home Directories
    read only = No
    browseable = No

[printers]
    comment = All Printers
    path = /var/spool/samba
    printable = Yes
```

Before putting this file in production, we must make sure that it does not contain any errors. To assist you in this task, Samba provides a tool called `testparm` to check parameter values. Entering `testparm` at the command prompt tests `smb.conf` and reports any problems.

If everything checks out correctly, restart `nmbd` and `smbd` using SWAT.

4.8.6 Creating shares

Now that we have properly configured an SMB server, we can create some shares for the benefit of users. These are the steps to create a shared directory using SWAT:

1. Open the Web browser and connect to SWAT. Click **SHARES**.
2. Enter a name for the share you want to create in the Create Share field. Click **Create Share**.
3. Fill in the comment and path fields with the appropriate information.
4. Click **Commit Changes**.
5. Click **Status**.
6. Click **Restart smbd**.

Now you can test the share you have created. First test it on the Linux server. Run this command:

```
[root@turbo samba]# smbclient -L localhost
```

You will be asked for a password. Type the SMB password you set for the root account. If you have not changed it since you last used `mksmbpasswd.sh`, the password will be blank. You should see a list of shares available on the Linux server, including the share you created.

To allow access by your Windows users, create user accounts on the Linux server for each Windows user with the `useradd` command. Then use `smbpasswd` to set the matching Samba password. This password should be identical to the one the user has to enter to log in to Windows.

For example, if our Windows user is `winuser`, with a Windows password of `123`, Example 4-5 shows the commands to add the user account:

Example 4-13 Creating an account for a Windows user

```
[root@turbo root]# useradd winuser
[root@turbo root]# smbpasswd -a winuser
New SMB password:
Retype new SMB password:
Added user winuser.
```

It is now possible to access this share from Windows. If prompted for a password, the user should enter the one used to log in to Windows.

4.8.7 Working with domain level security

To allow the Samba server to participate in a Windows 2000 domain as a member server, the steps below should be followed:

1. Stop `smbd` and `nmbd` on the Samba server.
2. Log on to the Windows 2000 domain.

3. Add a new computer account (the computer name should be identical to the NetBIOS name of the Samba server) in the Computers container of Active Directory Users and Computers, as shown in Figure 4-13:

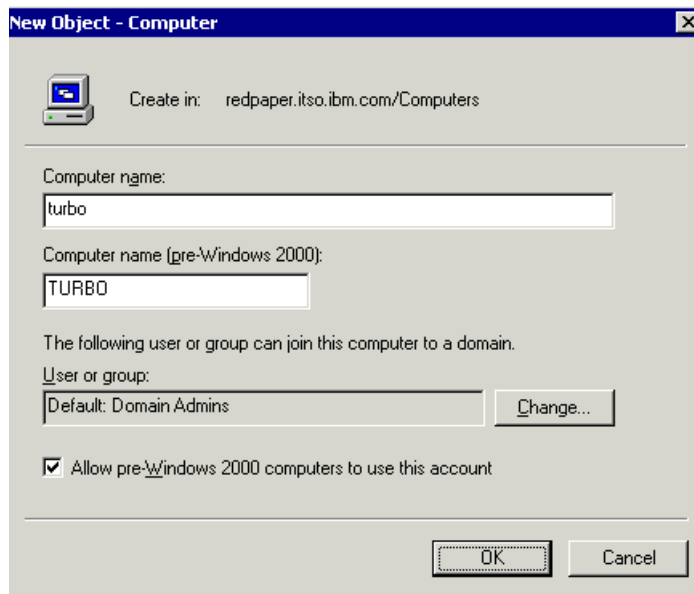


Figure 4-13 Creating a computer account for the Windows 2000 Domain

4. At the Linux server, enter the following command:

```
smbpasswd -j <NT_DOMAIN> -r <NT_PDC>,
```

where `NT_DOMAIN` is the Windows domain name, and `NT_PDC` is the computer name of the domain controller.
5. Access SWAT to modify the following parameters in the Globals section:

```
workgroup = NT_DOMAIN  
password server = NT_PDC  
security = domain
```
6. Start `nmbd` and `smbd`.

Now, when Samba receives a service request, it passes the user name and password to the domain controller for authentication. If the user name and password are confirmed to represent a valid user in the domain, and the user has been configured to have access to the requested resource in `smb.conf`, access is granted. Otherwise access is denied.

Samba will try to access files on the Linux server as per the client request, using the permissions set to the user account on the Linux server (a Linux user account for each domain user must exist; otherwise the access will fail).

4.9 Troubleshooting the integration process

In this section we discuss how to troubleshoot problems arising from the integration process. We discuss common network problems first, then move on to Samba-specific problems.

4.9.1 Common network problems

If you have connection problems during the integration process, one of the first things to check is that each of the computers involved has its IP address correctly configured. For Windows 2000 computers, the `ipconfig /all` command lists all of the relevant configuration information so that you can verify the TCP/IP configuration. On Linux computers, you can use `ifconfig -a` command to do the same thing.

For example, assuming you are using DHCP to allocate IP addresses in your network, these commands can help you determine if you are receiving correct IP addresses from the correct server, since they tell you the DHCP server's IP address.

If everything seems fine, use the `ping <IP address>` command to verify connections between computers. This command sends packets of data to the specified address. If all works correctly, the packet is returned, proving that there is a good connection between the two systems. Using a sequence such as the following takes you through a logical progression from verifying basic TCP/IP functionality, then your network adapter, and onwards, into the network:

1. `ping 127.0.0.1` to verify if basic TCP/IP function is correct.
2. `ping <address of localhost>` to verify if your network card is working properly.
3. `ping <address of gateway>` to verify you can get out to devices on the network.

On the Linux server, you should use `ping -n` so that the `ping` command does not try to find out the destination computer's host name. If you cannot receive a reply from the destination computer, please check all cable connections, gateway settings and network device status.

If pinging IP addresses works fine, the next to try is using host names (`ping <hostname>`) to verify name resolution functionality. If you are using a DNS server, please confirm that the correct resource records have been added to the DNS configuration. Keep in mind that, after you have modified the configuration, the DNS service must be restarted to ensure the new settings take effect. If you are not using DNS, every computer needs to have a correct hosts file. Name resolution is essential for both Windows 2000 and Linux systems; networking problems are often caused by an incorrect name resolution configuration.

4.9.2 Troubleshooting Samba-specific problems

As we have seen, setting up Samba is not particularly difficult. Difficulties can arise, however, so here we list some common problems, and offer tips and solutions that may help you to resolve them.

- ▶ If your Samba server is correctly configured, you will see one or two copies of `nmbd` running, along with a copy of `smbd` plus one additional copy of `smbd` for each connected user. You can check whether these services are running with the `ps ax | grep mbd` command.
- ▶ The most common problem when using Samba for the first time is that it cannot pass the password checking when accessing Samba from a Windows client. This is often caused by the global option `encrypt password = no`. You must set `encrypt password = yes` and then restart `smbd` to make Samba work with the latest release of Windows.
- ▶ After you have modified the Samba configuration, you must restart the SMB service to allow the new settings to take effect.
- ▶ If Samba is not working properly, try using another TCP/IP service, such as FTP, to verify that your network settings are correct. This can help you isolate the problem.

- ▶ If you cannot access SWAT from localhost, check the settings in your xinetd or inetd configuration files. Also make sure one of these two Internet services management daemon is running on your system (use `ps ax|grep inetd`).
- ▶ If you can access SWAT from localhost but not from another computer, check the security settings in the xinetd configuration file, `/etc/hosts.allow` and `/etc/hosts.deny`. If you compiled the Samba packages yourself, also make sure you added the correct configuration file in `/etc/pam.d`.
- ▶ Be careful when using “\” and “/”. Remember Samba on Linux always uses “/” in UNC and Windows uses “\”.

For other troubleshooting information, you can visit the Samba Web site at:

<http://www.samba.org>

4.10 Managing the integrated infrastructure

We close this chapter with useful information about implementing user accounts and permissions management.

4.10.1 User account creation and management

If you are using Samba to provide SMB services to your network clients, you must keep in mind that every user has to have an account on the Linux server to be able to access the Linux file system (unless you use share-level security). When users access the Samba server from a Windows client, the user name and password that were used to log on to Windows are passed to the Samba server. Typically the Samba server will validate the user name and the password first, through the local SMB password file (`security = user`) or will refer to another server (`security = server, domain`). If this validation succeeds, Samba uses the corresponding Linux account for that user name to access files on the Linux file system. So, in an integrated environment, you should create an account using the following steps:

1. Create a user account for Windows.
2. Create an account with the same name on your Linux server.
3. If you are using server-level security, add that account using `smbpasswd -a`, assigning the same password as for Windows.
4. Assign permissions to shares. For Samba shares, use SWAT.
5. Assign permissions for the file system.

However, there are other ways to create and manage user accounts, which make it easier in some situations.

Automatically create user accounts with `security = domain`

When your Samba server is operating as a member server of a Windows domain, the Samba service no longer uses the local SMB password file for user validation. Instead, the user name and the password are passed to a domain controller. If they are validated by the domain controller, the user is then allowed access to Samba resources. Samba also provides a mechanism to create accounts on the Linux server when a user is validated by the Samba server for the first time. So in a domain environment, it is possible to create user accounts for Windows and allow them access to your Samba servers.

This is very useful if you want each user to have a home directory stored on the network. A low-cost, fast and reliable Linux server is very suitable to hold these directories and fulfill other file serving functions. To do so, just connect to SWAT, go to the Globals section, and, in the Advanced View-Logon Options, set:

```
add user script=/usr/sbin/useradd %u
```

to automatically create an account when a user accesses the Samba server for the first time. If you also want to automatically create home directories for your users, set:

```
add user script=/usr/sbin/useradd %u -m
```

This creates a home directory for each user under the /home directory of the Linux server.

4.10.2 ACL and user name mapping

Windows NT and Windows 2000 use Access Control Lists (ACLs) for permission control, whereas the simpler file permissions of Linux are based on one owner and one group. Linux can also support an ACL, but currently it requires a lot of configuration effort and, worse, such ACLs cannot be backed up by backup tools. This is a management headache as it means that, should you have to restore files that have an ACL, you need to redefine every user's permissions. In a production environment, with many users, this is obviously unworkable.

Fortunately, Samba has a feature that can map Windows user names to Linux user names. If you need to group user access to some shares on a Samba server in domain security mode, this can be very helpful. For example, say you want build a Samba server and create a share sh1, and want Windows users u1, u2, and u3 to have full access to this share. This is how you would do this:

1. Create a user to represent the group, sh1full for example.
2. Create a directory sh1.
3. Connect to SWAT, make a share sh1, and add sh1full to the read and write list.
4. Use the **chown** command on the Linux server to make user sh1full the owner of directory sh1.
5. Make sure your smb.conf has a line like this in the global section:

```
user name map = /etc/samba.d/smbusers
```

6. Add the following line to /etc/samba.d/smbusers:

```
sh1full = u1 u2 u3
```

If you want to modify which Windows users have full access to sh1, just add or remove their names to or from the line in /etc/samba.d/smbusers.



Users

We close this document by changing our focus from your servers to your users and by taking a look at how an integrated Linux/Windows environment appears to them. In this integrated environment, users can access network resources provided by Linux servers just as easily as those provided by Windows servers.

Some examples are given to demonstrate how resources on Linux servers are accessed from Windows clients.

5.1 Browsing My Network Places

This is perhaps the most commonly used method for finding a network share. Browsing My Network Places helps the user locate shares on Linux servers as well as on Windows servers.

In our lab, we created a domain called Redpaper, which consisted of a number of Windows and Linux servers. Figure 5-1 shows the domain being browsed:

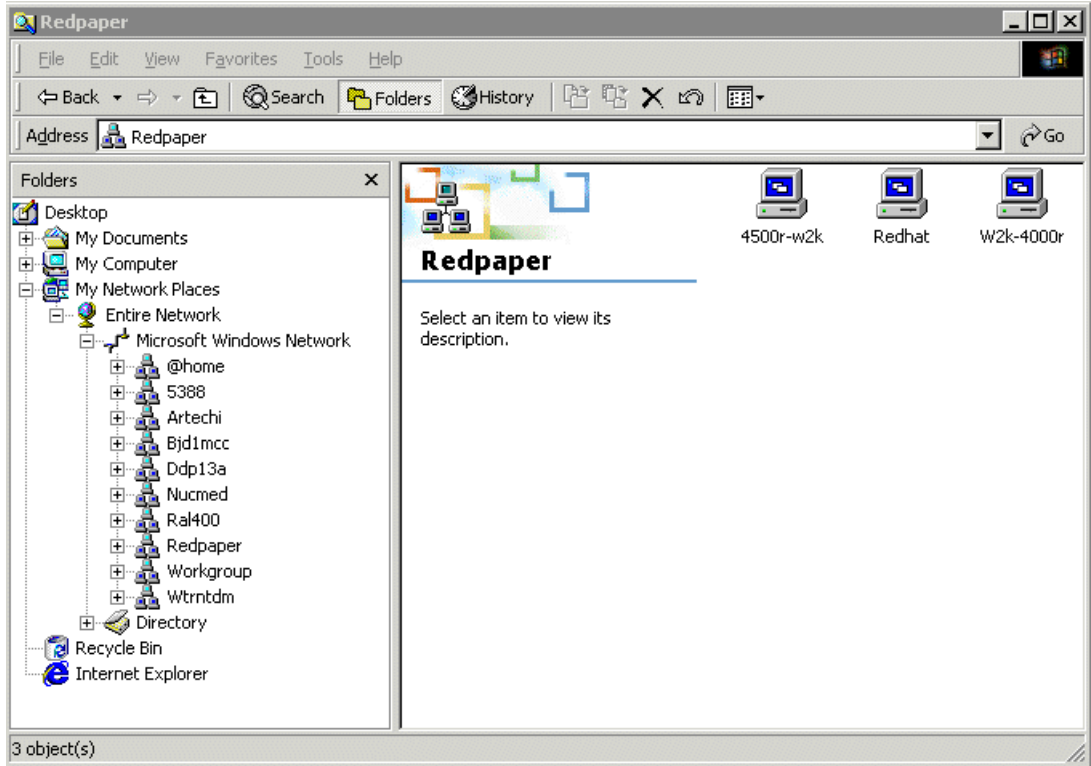


Figure 5-1 Finding a Linux server in My Network Places

In Figure 5-2, we are browsing the Redhat server and can see the shares, created by Samba, that are available on that system:

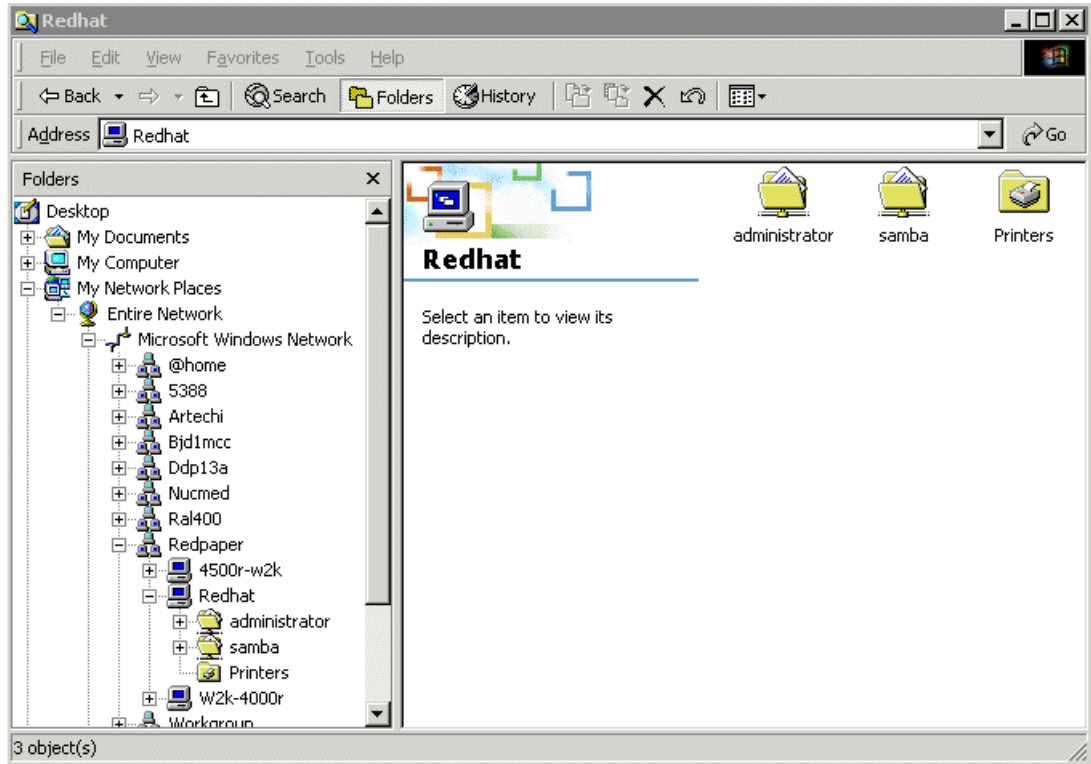


Figure 5-2 Accessing shares on the Linux server

5.2 Adding shares to Network Places

Users can also add shares on Linux servers to their list of Network Places, by using the Add Network Place Wizard as shown in Figure 5-3:

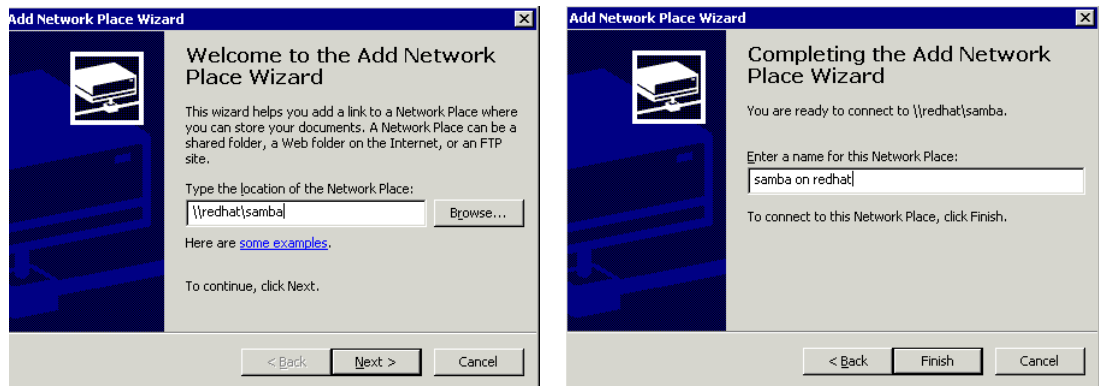


Figure 5-3 Adding a Network Place Wizard

Figure 5-4 shows the new Network Place:

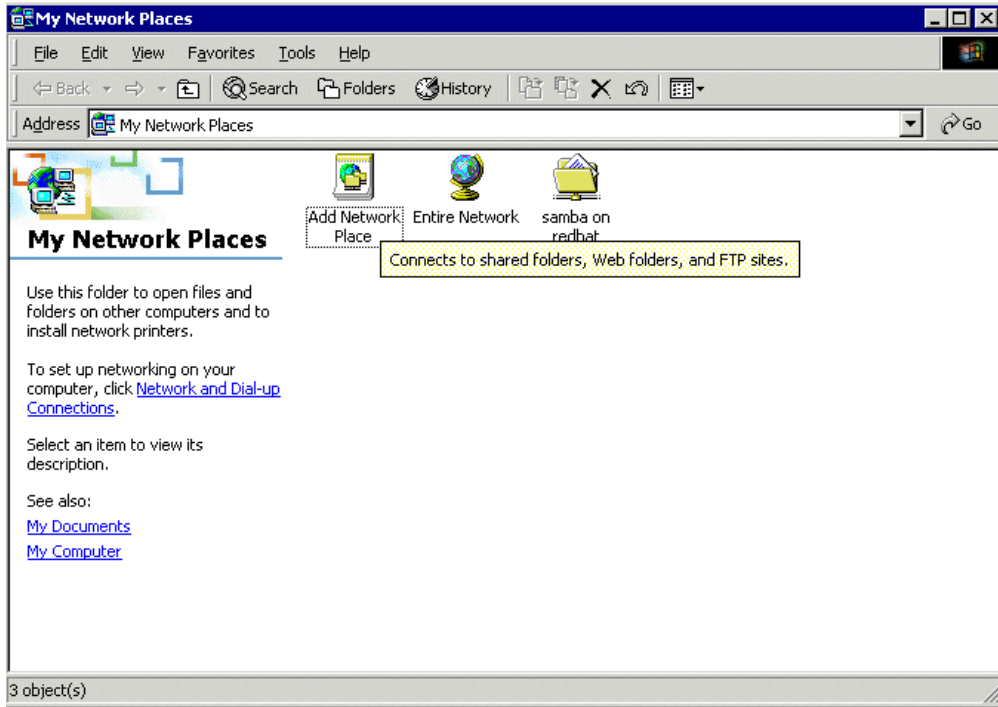


Figure 5-4 The added share in My Network Places

5.3 Mapping network drives

In addition to accessing Network Places, users can map their network drives, just as for a Windows share (see Figure 5-5):

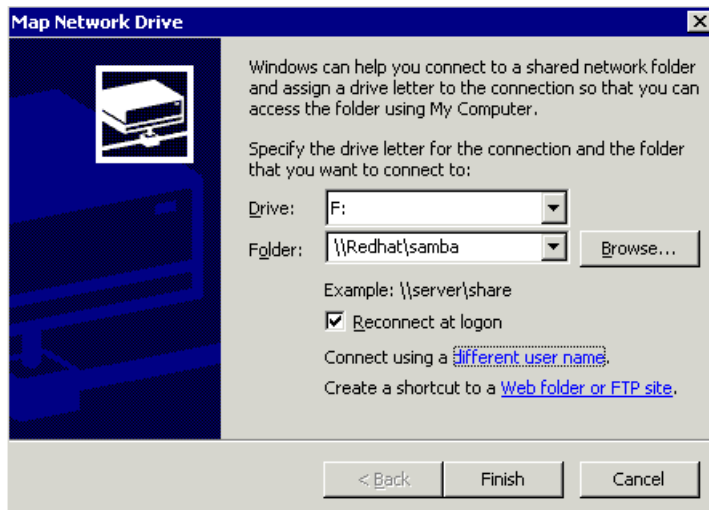


Figure 5-5 Mapping a network drive

5.4 Accessing Linux shares published in the Active Directory

Finally, Linux shares can be published in the Active Directory, so that users can access them by using the Directory Services. In Figure 5-6, the Linux share is published in the Active Directory:

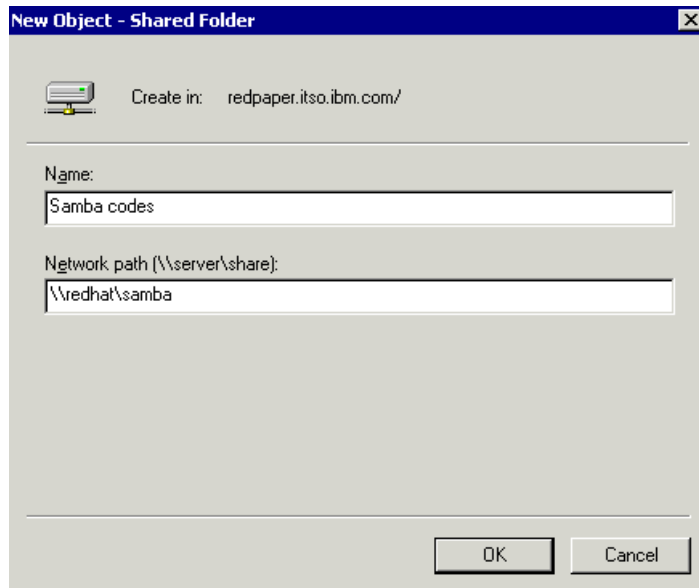


Figure 5-6 Publishing a Linux share in the Active Directory

Figure 5-7 shows how this can be accessed:

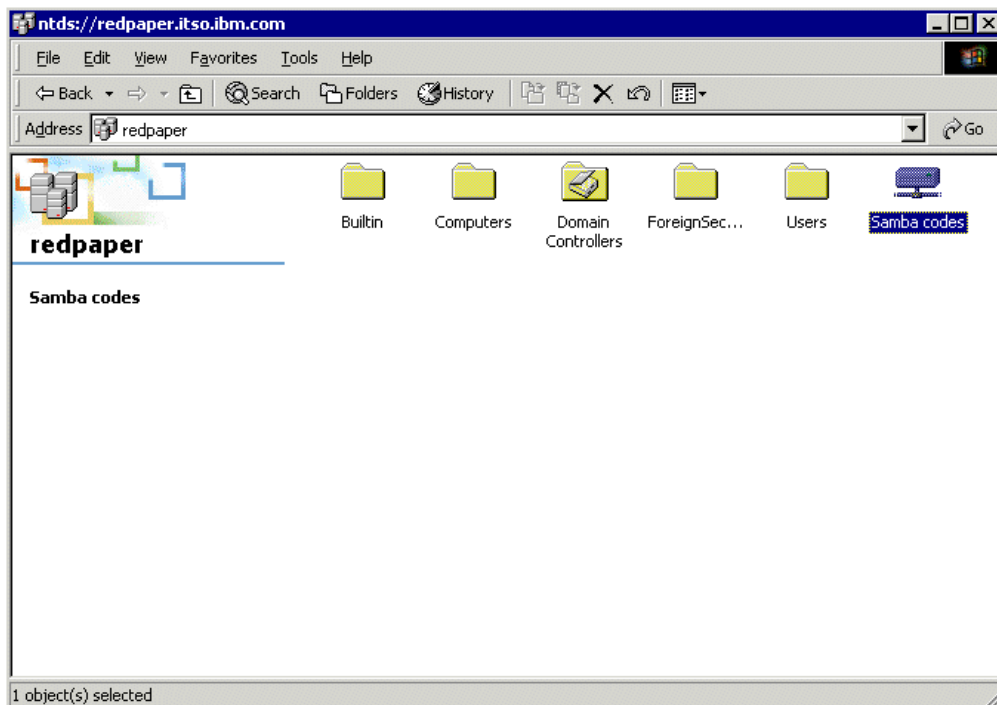


Figure 5-7 Accessing a published share in the Active Directory

These examples demonstrate that Linux network resources can be seamlessly integrated into your Windows infrastructure without the need for additional end user training. Integrating the two operating systems is not as daunting as it might originally seem and is actually very straightforward.



A

Useful references

This appendix is a compendium of the Web sites, Redbooks, and Redpapers to which we have referred throughout this document.

Web sites

- ▶ IBM IBM @server xSeries Web site:
<http://www.pc.ibm.com/us/eserver/xseries>
- ▶ White Paper: “Just what is SMB?” (V1.2, by Richard Sharpe, September 1999)
<http://samba.anu.edu.au/cifs/docs/what-is-smb.html>
- ▶ White Paper: “CIFS: A Common Internet File System” (by Paul Leach and Dan Perry, November 1996)
<http://www.microsoft.com/mind/1196/cifs.htm>
- ▶ Internet Engineering Task Force RFC Reference Library:
<http://www.ietf.org>
- ▶ Referenced RFCs:
<http://www.ietf.org/rfc/rfc1531.txt>
<http://www.ietf.org/rfc/rfc1034.txt>
<http://www.ietf.org/rfc/rfc1035.txt>
<http://www.ietf.org/rfc/rfc1001.txt>
<http://www.ietf.org/rfc/rfc1002.txt>
- ▶ SuSE Web site:
<http://www.suse.com>
- ▶ Red Hat Web site:
<http://www.redhat.com>
- ▶ Caldera Web site:
<http://www.caldera.com>

- ▶ TurboLinux Web site:
<http://www.turbolinux.com>
- ▶ Microsoft Web site:
<http://www.microsoft.com>
- ▶ Samba Web site:
<http://www.samba.org>

Referenced Redbooks

Redbooks are available from:

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

- ▶ *Red Hat Linux Integration Guide for IBM @server xSeries and Netfinity*, SG24-5853
- ▶ *Caldera OpenLinux Integration Guide for IBM @server xSeries and Netfinity*, SG24-5861
- ▶ *TurboLinux Integration Guide for IBM @server xSeries and Netfinity*,SG24-5862
- ▶ *SuSE Linux Integration Guide for IBM @server xSeries and Netfinity*, SG24-5863

Special notices

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, North Castle Drive, Armonk, NY 10504-1785.

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 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.

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively

through The Open Group.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.