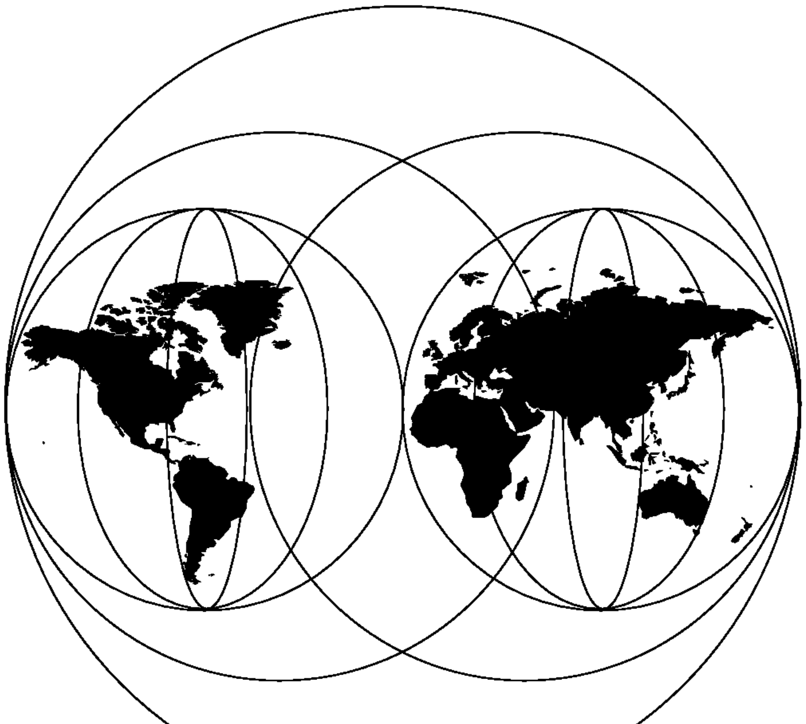
SG24-2021-00

Managing Access from Desktop to Datacenter: Introducing TME 10 Security Management

October 1997



International Technical Support Organization
Austin Center

International Technical Support Organization

SG24-2021-00

Managing Access from Desktop to Datacenter: Introducing TME 10 Security Management

October 1997

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix C, “Special Notices” on page 199.

First Edition (October 1997)

This edition applies to Version 3.2 of TME 10 Security Management. Discussion of the TME 10 Framework is primarily based around TME 10 Framework Version 3.1.

Comments may be addressed to (see also “Comments Welcome” on page xii):
IBM Corporation, International Technical Support Organization
Dept. DHHB Building 045 Internal Zip 2834
11400 Burnet Road
Austin, Texas 78758-3493

Contents

Figures

Tables

Preface

One disadvantage of the movement from the host-centric to the client/server and distributed model is the increased complexity of managing the systems involved. While the adoption of a network-centric approach may reduce or help to centralize some aspects of administration, providing secure user access to multitiered, multiplatform distributed environments becomes more difficult.

For those unfamiliar with complex security issues, this redbook provides a comprehensive discussion of security topics in the distributed network computing world. We demonstrate how a new TME 10 product, TME 10 Security Management, can help systems administrators reduce the burden of effective security management.

Besides looking at the TME 10 Security Management product itself, this redbook looks at wider security topics, explains security features already present in various system types, and looks at the general security features of the TME 10 environment provided through the TME 10 Framework.

Chapter 1 contains an introduction to the topics covered in this redbook.

The Team That Wrote This Redbook

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

Richard Hawes is an Advisory Systems Engineer at the International Technical Support Organization, Austin Center. Richard recently joined the ITSO from the European Software Project Office (based in the UK), where he provided technical troubleshooting and critical situation-management support on cross-platform software. Richard's most recent technical background is in OS/2 Warp Server and Windows NT.

Rolf Lendenmann is a Senior Systems Services Specialist in Zurich, Switzerland. Rolf started this redbook project while he was an Advisory Systems Engineer at the International Technical Support Organization, Austin Center. He has written extensively on Distributed Computing Environment (DCE) and the Tivoli system management products. Before joining the ITSO, Rolf worked for nine years in the AIX Technical Support Center in Zurich as a product specialist and consultant supporting all areas of AIX systems management, networking (TCP/IP, SNA), and middleware (DCE).

Lynn Holmes is a Senior IT Specialist in the UK. She has many years of experience in system management disciplines. She has worked with AIX for 10 years in various roles including technical support and solution integration. Lynn wrote sections of the redbook *TME 10 Cookbook for AIX*.

Gonzalo Quesada is an AIX Systems Specialist with GBM Costa Rica, an IBM Alliance Corporation. Gonzalo has nine years of experience in UNIX systems. He has worked with GBM for six years supporting the IBM RS/6000 family of products; he also has done consulting work on cross-platform environments involving relational database connectivity and network management. He has been part of previous residencies (DRDA/Datajoiner, AIX 4.2 and Using TME 10) at different IBM ITSO locations.

Alessandro Francalanci is a System Engineer with ISSC in Vimercate, Italy. His expertise has been in security and system management on the MVS platform. He is currently involved in the EMEA TE/SMP organization as Security Architect.

Development of a state-of-the-art security management product is a high-pressure task with many time constraints. The authors would like to thank the following people from Tivoli Systems, Austin, for taking the time to make invaluable contributions to this project:

Bill Kennedy	Gregg Wilson
Kevin Harper	Kate Brew
Jerry Frain	Rick Carlos

Comments Welcome

Congratulation or criticism, your comments are important to us!

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

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 213 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Home Pages at the following URLs:

For Internet users	`http://www.redbooks.ibm.com`
For IBM Intranet users	`http://w3.itso.ibm.com/redbooks`

- Send us a note at the following address:

  `redbook@vnet.ibm.com`

Chapter 1. Introduction

This chapter serves as an introduction of the main subjects that are discussed extensively in the following chapters. One of the major aims of this book is to provide you with an explanation of security-related terminology, first in a general sense, then as it applies to the types of systems commonly employed in distributed environments, and then more specifically as it applies to managing access to distributed IT resources.

1.1 Why Security Management?

Security in a computing environment has always been an issue, but now security in a networked and distributed environment is a growing concern for IT managers.

Previously, the large corporate environment was typically mainframe-based, where it was easier for administrators to secure data. It was usually one system, or maybe more than one, but at least they shared the same access control system, such as RACF in the IBM mainframe world. The security policy was largely dictated by the product, and it was all-encompassing. In addition, perhaps due to the relative complexity of the environment, the IT managers had more opportunity to develop an in-depth security culture.

It is no longer possible to deal with a single operating system and a single access control system. Businesses require their networked computing environment to share multiple platforms, each with its own security control mechanism (if they have one).

It is easy to understand why, from a security point of view, this new order is a big concern and why a lot of issues have arisen:

- Partner integration

  Software vendors are developing their own security tools, without any integration strategy.

- Security model based on specific platforms

  Every security application proposes its own security model; the security administrators are forced to have different approaches for different platforms.

- Some platforms don't provide an effective security tool.

  This is a weak link in the security chain of an enterprise.

All these issues open a gap between a company's security policy and what is actually in place or achievable. It is not uncommon that, to solve this sort of inconsistency, security policies follow a security model based on specific platforms. The result is weaknesses and inconsistency in security policies that become more serious the larger and more complex the organization.

Typically, security management has looked at different system types as isolated islands. There have always been difficulties correlating users, resources or events, such as access violations, among many systems. If security tools were able to manage more than one system, it was only because the systems shared the same platform.

Administration of assets is another major problem.

- Costs increase as the number of systems increases.
- Costs increase as the number of platforms increases.
- User satisfaction deteriorates as requests are not supported adequately.

The extra costs involved in adding new platforms to an IT infrastructure can limit flexibility and choice when it comes to selecting new systems. Often a selection can end up being a balance between the best product for the job and the easiest product to fit into the existing environment.

The challenge is how to manage the security of a networked computing environment with the associated complexity of applications. We want to merge the flexibility of distributed systems with the security of mainframe systems, at the same time managing the environment from a single point of control, with a single interface and with a consistent approach.

1.2 The Protection and Administration of Assets

We have already mentioned the term *security policy*. In order to effectively implement a security management product, it is important to understand the factors relating to the protection of network computing and other IT assets. It is also critical to stress the relevance of developing a company-wide security policy. Chapter 2 provides a discussion of security topics with ideas and methods designed to help the development of effective policies.

The security policy should establish the guidelines and people's responsibilities to protect an organization's information and other assets. It should establish, as well as guidelines and standards, the *roles* required to manage the security discipline and assign them the correct responsibilities.

1.3 Operating System Security

TME 10 Security Management brings together the management of complex security issues and policies over the whole enterprise. In the first release this encompasses UNIX and NT environments (and their desktop users), and soon this will grow to reach on up to the mainframe. In order to understand the implementation of security policy across a diverse collection of operating environments, it is helpful to know how each operating system platform implements its own security functions.

For Windows NT, TME 10 Security Management provides a way of centrally managing access policy on multiple workstations and NT domains. In the case of UNIX platforms, TME 10 Security Management provides major access to security enhancements through the use of Memco's proven Security for Open Systems (SeOS) technology in a component known as the Tivoli Access Control Facility (TACF). The same consistent interface is used for all platform types, and it is all integrated in the TME 10 Framework. This provides the consistent management of the same systems through other applications such as TME 10 User Administration or TME 10 Software Distribution.

Chapter 3 discusses what security exists in various operating system platforms and provides information on areas that require particular protection. The same information for soon-to-be-supported platforms, such as RACF, is in Chapter 8.

1.4 Security in the TME 10 Environment

When managing the security features of your network with products such as TME 10 Security Management, it's important to understand how the systems management environment itself is secured. If the integrity of the management system is compromised, then other aspects of network security come under threat, particularly when the management system is managing security!

The TME 10 environment, based around the TME 10 Framework, employs a number of security features to ensure the security management process itself is not subject to abuse. The notion of regions of responsibility is a fundamental part of the TME 10 Framework. Management tasks can be divided among administrators with little need for large numbers of all-powerful *superusers*.

1.5 TME 10 Security Management Implementation

TME 10 Security Management is a comprehensive product able to ease the complexity of managing security across heterogeneous environments and implement an enterprise-wide security policy. In the first release, TME 10 Security Management provides two major benefits:

1. Tivoli Access Control Facility (TACF). This is a sophisticated security product that gives a consistent implementation of access control across multiple UNIX platforms. It alone eases the management burden otherwise encountered when each platform has its own proprietary method. This product provides similar access control features to the MVS RACF product used in IBM mainframes and has many parallels with Windows NT security. It provides the added capability of removing the all-powerful nature of the UNIX *root* user.
2. A consistent, centralized management platform to control access to the supported UNIX platforms as well as Windows NT Domains and Workstations. TME 10 Security Management uses the popular TME 10 Framework paradigm of *Management by Subscription* and introduces the concept of *role-based access control*. The reach of TME 10 Security management extends beyond the directly supported platforms because it manages access to those platforms by other products. For example, by managing a Windows NT domain, we can manage the access to all domain resources, whether the clients are Windows 95, OS/2 or some other platform.

Soon after the publication of this book, TME 10 Security Management is to be extended to include host security management through links with products such as MVS RACF. In addition, the move to the Lightweight Client Framework (LCF) will provide further options for managing desktop systems.

1.6 TME 10 Security Management Installation

As a fully integrated TME 10 Framework application, installation on the variety of supported platforms is a relatively simple and consistent process. The product documentation provides details of the install process and we also provide a run-through of the install in Chapter 6.

1.7 TME 10 Security Management Usage

Chapter 5 goes into great detail about the implementation of TACF on UNIX systems and provides a very good description of the capabilities of TME 10 Security Management as it relates to UNIX security. In addition, we present in Chapter 7 some examples of scenarios designed to illustrate the implementation of TME 10 Security Management in real-world situations. For example, how do you prevent a UNIX root user from administering user accounts locally on a machine?

1.8 TME 10 Security Management Futures

As already mentioned TME 10 Security Management is going to expand rapidly into other areas to really enable management of IT asset access from the desktop to the datacenter. In a similar format to earlier chapters, Chapter 8 discusses similar topics relating to MVS RACF and to other platforms such as Novell NetWare and OS/2 Warp Server.

Other plans for the future of TME 10 Security Management include closer integration with the TME 10 User Administration product. These two can be used independently of one another, but there are many opportunities for sharing data and tasks between them.

1.9 Additional Topics

The main subject of this book is the TME 10 Security Management product. It would be impossible to cover all aspects of computer security in a single book, but in discussing security issues, it is helpful to have some descriptions of commonly referred-to security concepts such as authentication, authorization and encryption. Appendix A, “General Security Concepts and Terminology” on page 189 provides those descriptions and gives quick definitions for various security-related terms.

Any discussion of the TME 10 environment would be incomplete without some mention of the Lightweight Client Framework (LCF) implemented from TME 10 Framework Release 3.2. LCF is a major scalability enhancement for TME 10 and, almost every TME 10 product will be affected by it. Appendix B, “Introduction to TME 10 Framework Version 3.2” on page 195 provides some background on this topic.

Chapter 2. Security Concepts and Overview

The objective of this chapter is to discuss the basic concepts of security in a networked computing environment. This provides the basis for implementing effective security management. Before we can deploy software, such as the TME 10 Security Management product, that helps protect your company's information assets from unauthorized access, the following must be in place or be defined:

- A security policy
- Security project components:
  - Specifications for physical and logical security
  - Roles and responsibilities
  - Events that lead to violations of policies and how to react to them through auditing, reporting and alerting

A discussion of these issues will help you understand what the TME 10 Security Management product can do for you and what the activities are that should precede a successful installation and deployment. The chapter concludes with an introduction of an organization-based security management model.

2.1 Security Policy

Before discussing any type of security implementation, it must be stated that security policy is a business-related issue. The objective of a security policy is to protect the company's assets, such as real estate, equipment, safety of employees, and information. For many companies, information is one of the most crucial company assets: If you think of an insurance company, the loss or alteration of information really could mean the loss of business, and the existence of the company could be at stake. Management must be aware of the importance of protecting the company's assets, in particular the information assets. The most effective way to show its awareness is to issue and stick to a company-wide security policy. The security policy should be signed by the top-level management, and it must apply to all employees.

What is expected from management of the organization is to:

- Define a security policy.
- Evaluate the risks and define the needed budget.
- Enforce the security policy.

Good management control must be in place to have a solid base on which a good technology can rely. Without a security policy in place, the information security cannot be implemented effectively, not even with a state-of-the-art product, such as TME 10 Security Management. Every project leader involved in security projects, in particular with TME 10 Security Management, should be aware of this crucial prerequisite for the success of the project.

2.1.1 Security Threats

Before implementing a security policy, the management should assess the risks their computing environment is exposed to and take the appropriate actions; the obvious consideration is that it is not worthwhile to spend more money in controlling threats than the expected loss if no action were taken.

Generally speaking, the computing environment is exposed to two main types of risk:

1. Loss or alteration of data
2. Discontinuation of service

It is not easy to determine the origins of the threats. They can vary from unauthorized disclosures to viruses, but human error still is the most common security threat, as shown in the following figure.

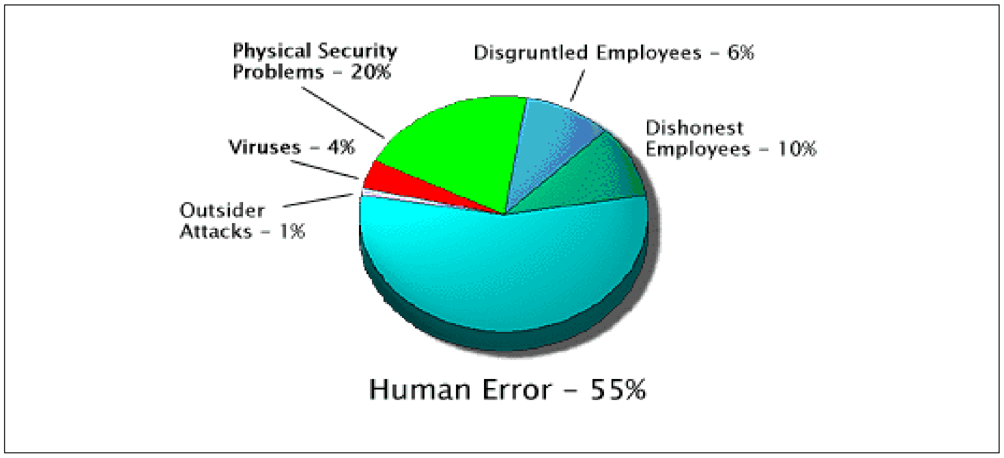


Figure 1. Sources of Security Violations

This statistic can help us to address our effort to effectively decrease the security threats to the computing environment. For example, attacks from outside the company should not be ignored, but looking at ways to reduce the impact of human error should be a major focus area.

2.1.2 What Should a Policy Include?

The definition of an organization-wide policy regarding the protection of information is the first, and probably the most important, action to be taken.

A computer security policy should establish the guidelines and responsibilities to protect the information assets of a company. In particular, a security policy should define:

- ***Standards*** **-** These define the criteria security management must follow. Standards are usually split into two main areas:
  1. Physical access controls - locking servers away, storage of backups, and so forth.
  2. Logical access controls - User IDs, administrator roles etc.
- ***Rules*** - These specify the guidelines for conducting security management. The most common rules are the rules for maintaining passwords, for example.

  It is a big help for the involved professionals, such as the security administrator, if the guidelines specify how the different operating system platforms could address the specific rules.

  With TME 10 Security Management, the enforcement of common rules is much easier.

The TME 10 System Policy Record allows the definition of systemwide rules, such as:

- Audit control
- Login policy
- Password policy
- Resource type policy

The benefits are, at least, twofold: First, the consistency of rules across the systems is improved, and second, it is not required that the security administrator have a specific knowledge of the different system security services.

- ***Processes*** - A process should specify how a standard has been implemented.

  Typically, a process is composed of one or more activities, where every activity consists of four parts:

  1. An input, for example the request to reset the password.
  2. A mechanism that indicates the means or the roles by which or by whom the activity is performed.

     In other words, it determines what roles are involved in this specific activity. In our example, the roles could be the user who requires the new password and the security administrator.
  3. A control that describes the condition or the circumstances that govern the activity.

     For example, the standard for the request of a new password may specify that the requestor must be positively identified.
  4. An output, which is the result of the activity.

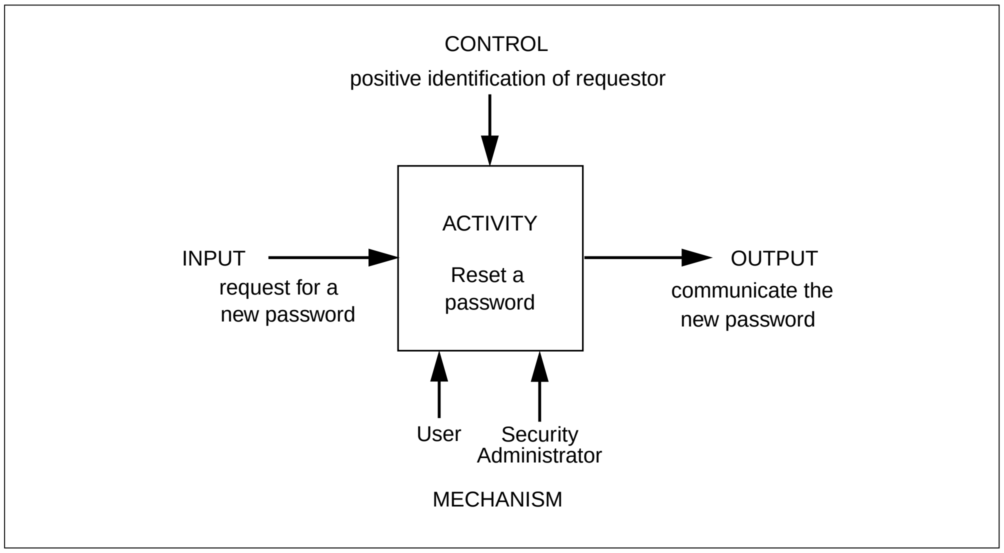

CONTROL
positive identification of requestor
INPUT
request for a new password
ACTIVITY
Reset a password
OUTPUT
communicate the new password
User
Security Administrator
MECHANISM

Figure 2. Graphical Description of an Activity within a Security Policy Process

By linking all the activities, we obtain the final process and the identification of the roles needed to execute the process itself. In the example, the process is composed of one activity; the roles identified are the User and the Security Administrator.

Standards, rules and processes must be documented for auditing purposes, and a mechanism should be in place to assure their currency.

Moreover, a security policy should:

- Define the security-relevant roles
- Assign responsibilities to the roles

As previously stated, the roles are identified during the process definition. The policy applies to all employees, and management must ensure employee awareness.

2.2 Security Project Components

In this section, we discuss the implementation of security. We are not yet talking about deploying a security management product, we are still working on the design of a security project. The topics that follow should constitute the functional specifications of a security project.

2.2.1 Physical Security

Physical security - the protection of resources through their location or through hardware devices, especially when the population of the physical area where the system components reside is very heterogeneous. Multi-company people in the same building, such as customers, subcontractors or students, are the most common situations that require physical protection of a computing environment.

The method to implement physical security is to identify, first of all, the components that need to be physically secured. Such a list should include:

- CPUs and system units
- LAN infrastructure components, such as LAN management systems, bridges, routers, wiring hubs, active ports, and performance and trace tools
- LAN-connected systems
- Storage media, such as magnetic tapes, disks and cartridges

Then, you should identify two or three areas with different levels of security classification such as:

Public	Areas that all employees can access
Controlled	Areas that can and must be locked when unattended
Very controlled	Areas where access is restricted to registered authorized users

Finally, it is possible to build a matrix where every area is assigned to a single system component or a topology of system components, according to their security sensitivity.

2.2.2 Logical Security

In the following sections, we discuss what the most relevant issues are when you implement your own logical security system. By logical, we mean those aspects that rely on definition and maintenance of system users, access rights, policies, and so on.

2.2.2.1 Identifying and Authenticating Users

Identification of users is the very first step in implementing a logical security system. This requirement should be addressed by defining a user ID in the access control system. The user ID identifies the person connecting to the system and ultimately what resources that person has access to. As far as possible, a user ID should be accountable to an individual; the sharing of user IDs should be avoided unless this is done in a controlled manner under the responsibility of the owner of the user ID.

Checking a password is still the most common method to authenticate users; that is, the password is used to verify that users are who they say they are. The robustness of password quality is the key to enforcing effective user authentication control.

2.2.2.2 Defining and Protecting Resources

Resources (files, databases, programs, data) can be divided into two main subgroups:

1. ***Operating System Resources*** - Operating system resources are all those data objects related to system services or functions; they include system programs and files, subsystems and program products.

   Operating system resources are usually under the control and responsibility of the provider of service. Their integrity must be assured since the data is critical for the service an organization wants to supply.

   Operating system resources are not always read-sensitive, although a list of exceptions must be identified and protected accordingly. A typical example of an exception is the database where user IDs and passwords are stored.

2. ***User Resources*** - User resources are all those data objects related to individuals or groups. User resources should be protected according to the data owner's prescriptions (see Section 2.2.3, "Roles and Responsibilities in Security" on page 10, for definitions of terms such as data owner). It is advisable to set an initial default protection in order to guarantee a minimum level of security.

2.2.2.3 Defining the Administrative Authority

Among all the user IDs that populate a computing environment, some of them have particular authorities; the user IDs need these authorities to administer the computing systems.

We can differentiate the administrative authority in two different categories. It is very important to understand the difference, since the criterion used to assign them and to periodically verify their business needs can be very different.

1. ***Security Administrative Authority*** - This type of authority allows the administrator to perform those actions needed to administer security. This authority may allow the administrator the permission to alter system components or to read confidential data. It must be considered a misuse of authority if such an action is taken when there is no appropriate business need.

2. ***System Administrative Authority*** - This type of authority allows the administrator to perform all those actions needed to administer the system. This authority might allow the administrator to bypass the security control, but it must be considered a misuse of authority to do so.

System and security administrative authorities are both security-relevant. What should be done now, is to:

- Define for every system platform or access control system the authorities that can be recognized in the above two categories.
- Assign the authorities according to the individual responsibilities.
- Periodically review the assignment of the authorized user IDs and verify their business need.

2.2.3 Roles and Responsibilities in Security

As you may have realized from the discussion of process mechanisms in Section 2.1.2, “What Should a Policy Include?” on page 6, many roles could be defined. This depends on how many processes a company has in place.

The same types of roles can be found in many organizations. Speaking of security, for example, the most relevant role is the provider of services. Following is a list of commonly defined roles:

- ***Provider of Services*** - The organization/group and/or manager that is providing information processing services. Usually, this organization is responsible for ensuring the security of the computing environment.
- ***Data Owner*** - The manager responsible for controlling the security of assigned data. The data owner’s responsibilities are to:
  - Evaluate the sensitivity of data in order to classify them
  - Determine the required level of protection (according to the classification)
  - Approve or deny a request for access to data under his or her responsibility

  The data owner role can reside in any organization, including the provider of services.
- ***Auditor*** - The auditor is the person responsible for:
  - Enforcing the security policies
  - Enforcing the security processes
  - Periodically performing a control assessment
  - Setting requirements for applications/tools/solutions to comply with security needs
  - Defining an adequate disaster recovery plan

  Typically, the auditor role resides in the organization of the provider of services.
- ***Security Administrator*** - The security administrator is the person responsible for setting and administering systemwide security controls. Security administrator responsibilities are to:
  - Provide the system security setting as defined in the standards and rules. (Using the TME 10 terminology, the security administrator is responsible for setting up the system policies, including password policy, audit policy, login policy, and default access to resources types.)
  - Provide system access to the users through the definition of new user IDs, resetting of passwords and removal of old user IDs
  - Perform periodic controls in order to verify the security status of the computing environment

Typically, the role of security administrator resides in the organization of the provider of services.

- ***Security Officer*** **-** The security officer is the person responsible for the security administration within the business of a data owner. The security officer responsibilities are to:
  - Provide data security settings as specified by the data owner, including defining the audit policy and the default access to their own data. These definitions are made in addition to the system policy to better meet the data owner's needs
  - Define access permissions to data according to data owner approval

  Typically, the role of security officer resides in the data owner group. However, in less structured organizations, the role and its responsibilities can be assigned to the security administrator.
- ***Data User*** **-** the data user's responsibilities are to:
  - Follow the security instructions, even if the system control doesn't force them to. For example, the password must not be trivial and must follow the approved syntax rules. This must apply on every system, independent of the security control in place.
  - Use their access authorities, system authorities or administrative authorities only for management-approved use.

  Every user of the computing environment is a data user, regardless of the group to which they belong.

2.2.4 Auditing and Reporting

With the terms *Auditing* and *Reporting* we mean the ability to record all the security-relevant activity performed in the computing environment. Coupled with this is the ability to report this activity in a human-readable form.

A good auditing setup is very important in security; it must always provide a clear picture of security status. Moreover, it constitutes a powerful passive *agent*. We have seen in 2.1.1, "Security Threats" on page 5, that 16 percent of the security threats are due to disgruntled or dishonest employees. Effective audit trails of activities is a strong deterrent to those types of threats.

There are two main things to keep in mind when an auditing policy is to be implemented:

1. Decide what events are really important for security. Recording everything is not a good choice but a waste of disk space and can cause many problems when a report is to be generated. For certain audit items, it may be that not every activity is logged or included in reports; it may be that some statistics are dependent on certain thresholds being met. Here is a recommendation of a minimum list of events to log:
   - All security violations, such as:
     - System access denied
     - Invalid password
     - Password revoked
     - Resource access denied
   - All accesses to sensitive/significant areas of the systems

- All security commands issued using the security administrative authority
- All accesses to operating system resources, with the exception of the default access. If the resources can be read publicly, we need to record the update or allocation and deletion of access.

2. Decide for how long the records must be stored and define a storage plan accordingly.

TME 10 Security Management can help you in different ways to reach these objectives:

- Although auditing itself is performed by system-specific security services (for instance, TACF/UNIX, NT and so forth), the Tivoli Management Environment helps you to establish a consistent auditing policy across your systems. This is achieved both through the use of TME 10 Security Management facilities and through other TME 10 products such as TME 10 Distributed Systems Monitoring (previously known as TME 10 Sentry).
- TME 10 Security Management provides a Security Notice Group as a collection and auditing point for tracking all the TME 10 Security Management application-related operations. In other words, the Security Notice Group is the means to inform the TME Security Auditor about the TME 10 security-relevant operations. These would include the creation of a new security profile, the definition of a new role, group or resource, or a new system policy.
- TME 10 Security Management provides the Audit Report Generation Task. This task allows the TME 10 Security Auditor to gather security information in a logical fashion from different endpoints and to produce a report for audit purposes.

2.2.5 Alert Management

In this section, we want to stress the importance in security to have the ability to react immediately when the computing environment is supposed to have been exposed to a danger, such as the danger of an attack to the system in an attempt to gain unauthorized access. The objective is to detect, in real time, when the danger occurs and issue an alert.

A sample of the process to follow in order to detect the problem and issue an alert could be the following:

- Every security violation should generate a system event.
- A single system event is not necessarily sufficient to say this is a danger; the events could be collected in such a case.
- The collection of events should then be compared with predefined thresholds.
- If the result of the collection exceeds the threshold, an alert MUST be issued.

The result of this process is that we can ignore someone typing in a wrong password on Monday morning, but we must not ignore someone else entering many wrong passwords associated with many user IDs on a Saturday evening!

Also important for the alert management is a correct definition of roles and responsibilities and the assignment of those roles and responsibilities to the appropriate managers. The objective of good alert management is, in addition of detecting the alert, to notify the appropriate persons who know exactly what to do.

The Tivoli architecture is vital in this area. TME 10 Security Management has the extraordinary capability to detect different events from different endpoints with different platforms. Thanks to the specific security service (for instance, TACF, NT) Audit Event Adapters, the TME10 Enterprise Console Logfile Adapter and the TME 10 Enterprise Console, it is possible to generate audit events and collect them according to predefined rules. Then either an alert can be issued, or some automated process can be initiated.

2.3 Definition of a Security Management Model

In many computing environments, the funding for security is mainly used for the administration of user IDs and the administration of the accesses to the resources. These activities are often clerical and repetitive activities. They are needed to satisfy the users' requests and to supply the committed level of service, but a big effort in users and resources registration doesn't mean that security is automatically assured. In fact, as we have seen in the previous section, good system security can only be obtained if a good security policy is in place. This includes having security roles well defined and control and auditing operations being performed daily.

The design of a security project should take care of these considerations. It should propose methods and technologies consistent with the security policy. It should enable a decrease in the pure administrative cost while addressing some effort toward more effective security operations.

2.3.1 Security Administrative Model Overview

The most efficient method to reduce the administrative cost is to propose a security administrative model based on *job functions* and *user groups.*

2.3.1.1 Job Functions

A Job Function description defines a set of capabilities required to carry out a given business activity. In our scope, it describes the resources (files, databases, program, terminal) a job function should access with a defined authority.

A job function description answers the question: What can I do to what resources?

TME 10 Security Management calls the job function description a *role* (which although they may coincide, should not be confused with the concepts described in "Roles and Responsibilities in Security" on page 10).

2.3.1.2 User Groups

A user group defines a set of users with similar responsibilities inside the organization. In our scope, it describes the users requiring the same authorizations.

If the groups are associated with job functions, the membership to different groups answers the question of any particular user: What is my function?

If we are now able to associate user IDs to user groups and user groups to job functions (roles), we have designed an efficient model for administering security.

TME 10 Security Management also uses the term *group* to refer to a set of users performing similar functions. Users may, of course, belong to more than one group, and each group is almost certain to contain multiple roles.

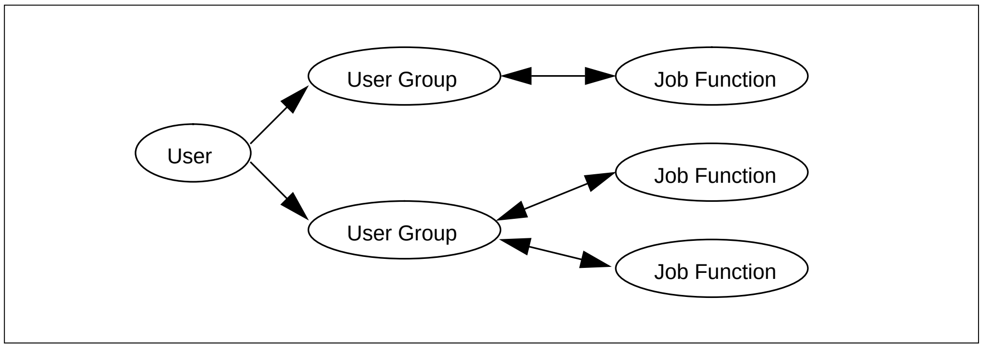


Figure 3. A Security Model

We don't need to administer the user's accesses one by one. We only to manage the associations, which is much more efficient. Adding a user to a group, moving from one group to another or removing from groups altogether ensures the correct access is defined for that user to the required collection of resources. Those resources can be spread across many different systems, even different platform types.

2.3.2 Implementing an Organizational Approach

The model proposed in the previous section "Security Administrative Model Overview" on page 13 can be addressed in different manners, depending on the security service installed. As stated in that section, TME 10 Security Management takes a role-based approach. Here, we discuss the differences between the role-based approach and a more traditional group-based approach.

2.3.2.1 Group-Based Approach

An approach based on groups defines at the resource who may access it and with what permissions. A popular implementation of this approach is to create access control lists (ACLs) on the objects. The ACLs specify users and groups with their access rights to the objects.

This is very common among the current security services. For example, UNIX, RACF and NT use this approach. When you use this type of approach, you have to define groups, add them to the objects as grantees of access, and then connect the user IDs to the groups. This approach works well, but is very product-oriented and technical, with little relationship to a company's structure.

2.3.2.2 Role-Based Approach

TME 10 Security Management uses a new approach; it abstracts from the access control mechanism implemented at the endpoint, and it introduces the concept of roles. The security administrator defines roles and assigns object access permissions to each role.

Since a specific type of job can encompass multiple roles (or job functions) and multiple job types may have overlapping responsibilities (roles, job functions), TME 10 Security Management also uses the concept of groups in addition to roles. However, these groups in conjunction with roles are a more organizationally-oriented concept. Users are assigned to groups according to the job they have to perform, and groups are assigned roles according to the job functions that a particular type of job has to perform.

This is similar to the *groups-within-groups* capability of the Windows NT environment, expanded to a cross-platform, organization-centric view.

The first benefit from this approach is that the security administrator works with more familiar, real-life concepts, like the organization of his or her company and its related needs. Once configured, TME 10 Security Management then takes care of mapping the organization description into appropriate, system-specific endpoint definitions. The security administrator doesn't need to know specific commands for the different security services any more. TME 10 Security Management takes care of it. This is a major benefit for administrators.

In Chapter 5, “Understanding TME 10 Security Management” on page 57, the use of roles and groups in TME 10 Security Management is described extensively, with particular examples of how TACF implements this in the UNIX environment.

Chapter 3. Security Implementations on Different Platforms

This chapter consists of two major parts. First, we discuss security features and concepts already provided by the various operating system environments to protect their resources. We then highlight which resources should be particularly protected. We discuss these issues for the following operating systems:

- UNIX
- Microsoft Windows NT

TME 10 Security Management will very soon be supporting OS/2, Novell NetWare and MVS with RACF. Refer to Chapter 8, "TME 10 Security Management Futures" on page 163, for similar details of these environments.

Note:

When we discuss operating environments, particularly Windows NT, we are really talking about access to the resources contained within that environment. The users of those resources may extend to many other platforms, including OS/2 and other flavors of Windows.

Many additional access control products exist for UNIX environments. This chapter describes the basic protection provided by the leading UNIX flavors (as well as Windows NT). TME 10 Security Management provides consistent management of resource protection on UNIX platforms through the use of TACF, described in Chapter 5, "Understanding TME 10 Security Management" on page 57.

3.1 Security Structures

Most operating systems provide some form of basic protection of data and hardware physical resources through the implementation of security structures and authorization mechanisms.

3.1.1 UNIX Platforms

The basic security features provided on all UNIX platforms are: authentication of users through password verification and access control using the permission bits on files for owner, group, and users (users being all others).

In addition to these standard features, many UNIX implementations or flavors have implemented extended security features, such as tighter authentication methods, access control, trusted computing base, and auditing. After discussing some general UNIX mechanisms, we need to take a closer look at some different UNIX flavors. Although these extra features follow standards, their implementation is different, thereby making them proprietary and incompatible. In Chapter 5, "Understanding TME 10 Security Management" on page 57, we discuss a solution that allows you to replace all the proprietary features and create common implementations of security policies. We limit the discussion of security features in this chapter to the major UNIX implementations: AIX, SunSoft Solaris, and Hewlett Packard HP-UX 10.

3.1.1.1 UNIX UID/GIDs Issues

The area of security in the UNIX operating system environment has been a topic of discussion lately, since UNIX was not developed with security in mind. UNIX can be weak in protecting against crashing or crippling the operation of the system.

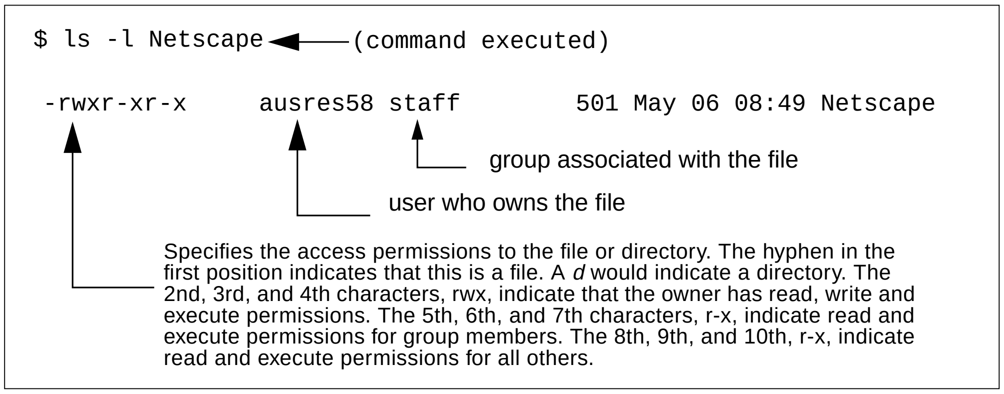
In UNIX each file has associated with it eleven bits of protection information together with a user identification number and a user-group identification number (UID and GID). Nine of the protection bits are used to specify permissions to read, to write, and to execute the file for the user (owner), to members of the user's group, and to all other users accessing the system.

All UNIX systems contain a form of discretionary access controls. These controls are of the form:

```
-rwxrwxrwx        user        group        filename


where first position (-) is one of  d directory
                                    c character
                                    s special
                                    b block
                                    - regular file
                    next positions
                                    r is read
                                    w is write
                                    x is execute
```

Each file has a filename, an owner (the user) and an associated group. The first character in the -rwxrwxrwx specifies what type of file it is; the next three indicate the access rights of the user; the middle three indicate the access rights of the group, and the last three indicate the access rights of all other users on the system. A hyphen (-) in any of the last nine positions indicates that access right is *not* allowed for that particular type of user.

The following example shows access permissions for the file Netscape:



```
$ ls -l Netscape

 -rwxr-xr-x     ausres58 staff        501 May 06 08:49 Netscape
```

Specifies the access permissions to the file or directory. The hyphen in the first position indicates that this is a file. A *d* would indicate a directory. The 2nd, 3rd, and 4th characters, rwx, indicate that the owner has read, write and execute permissions. The 5th, 6th, and 7th characters, r-x, indicate read and execute permissions for group members. The 8th, 9th, and 10th, r-x, indicate read and execute permissions for all others.

Figure 4. File Access Permissions for a Specific File

A process generated by or for the user has associated with it an effective UID and a real UID, and an effective and real GID. When an attempt is made to access the file for reading, writing, or execution, the user process's effective UID is compared against the file's UID; if there is a match, access is granted provided the read, write, or execute bit, respectively, for the user himself is present. If the UID for the file and for the process fail to match, but the GIDs do match, the group bits are used; if the GID's do not match, the bits for the users are tested.

The last two bits of each file's protection information, called the set-UID and set-GID bits, are used only when the file is executed as a program. If the set-UID bit is on and that file is executed, the process runs under the effective UID associated with the file rather than the one associated with the calling process or user.

Similarly the effective group ID of a process is changed to the GID associated with a file when that file is executed and has the set-GID bit set. The real UID and GID of a process do not change when any file is executed, but only change as the result of a privileged system call.

On the issue of password security, UNIX is very solid. All passwords are stored in an encrypted form under the /etc/passwd file (usually). These passwords are encrypted with a modified version of the Data Encryption Standard (DES) so that it becomes a one way crypt. When a user logs in, the password entered to the operating system at the `password:` prompt is encrypted using the modified DES routine and compared against the encrypted password inside of the /etc/passwd or /etc/security/passwd file. If the two match, the user is allowed to log in to the system.

To reduce the effectiveness of key search (a form of security threat), a defense mechanism called *password salting* is used. This mechanism employs a random 12-bit number, the salt (refer to Table 1 for an example), which is appended onto the password when the password is first entered into the system. This string, the 12-bit number, and password are encrypted and stored in the password file. When a user attempts to log in, the salt, stored in the password file, is appended to the supplied password, encrypted, and compared to the encrypted string. If the two strings match, the user is allowed to log in.

Table 1. Passwords and Salts

Password	Salt	Encrypted Password
orange	77	777EXTC2hioog
bluecat	d8	d8tLsKnYEeqgo
oyster45	t9	t9pQTTXKQjKN.

However, some UNIX platforms save the sensitive data, such as encrypted passwords, in a read-protected /etc/security or shadow file/directory, and access is redirected from the standard files to the secure files on a field basis.

3.1.1.2 IBM AIX

AIX security is based on establishing and maintaining proper access control and accountability policies. The security administrator of an AIX system is responsible for configuring the following aspects of security:

- Managing protected resources with access control

Access control involves managing protected resources using the setuid and setgid programs and hard-copy labeling (user and group names). The operating system supports several types of information resources, or objects.

The most important types of objects are:

- Files and directories (used for information storage)
- Named pipes, message queues, shared memory segments, and semaphores (used for information transfer between processes)

Each object has an associated owner, group and mode. The mode defines access permissions for the owner, group and other users.

- Access Control Lists

  Access control is composed of protected information resources that specify who can be granted access to such resources. Basically, the major task in administering access control is to define the group memberships of users, because these memberships determine the users' access rights to the files they *do not* own.

  Access control lists (ACLs) increase the quality of file access controls by adding extended permissions that modify the base permissions assigned to individuals and groups. With the use of extended permissions, you can permit or deny file access to specific individuals or groups without changing the base permissions.

  Access control lists are maintained by the following commands:

aclget	Displays the access control information of a file
acledit	Edits the access control information of a file
aclput	Sets the access control information of a file

- User Account Control

  Each user has a set of associated attributes. These attributes are created from default values when a user is created for the first time. All of the attributes are defined in the /usr/lib/security/mkuser.default, /etc/security/user, /etc/security/limits and /etc/security/lastlog files.

- Identification and Authentication

  Identification and authentication establish a user's identity. Each user is required to log in to the system. The user supplies the user name of an account and a password (in a secure system, all accounts should either have passwords or be invalidated). If the correct password is provided, the user is logged in to that account; the user acquires the access rights and privileges of the account. The /etc/passwd and /etc/security/passwd files maintain user passwords; the /etc/passwd contains a bang(!) for the password that redirects access to the /etc/security/passwd file.

- Password Restrictions

  AIX provides configurable password restrictions; these characteristics enable the administrator to constrain passwords chosen by users and to force passwords to be changed regularly. These restrictions are recorded in the /etc/security/user attribute file and are enforced whenever a new password is defined for a user.

  The restrictions that can be applied are:

  - *minage:* minimum number of weeks that must pass before a password can be changed.

  - *maxage:* maximum number of weeks that can pass before a password must be changed.
  - *maxexpired:* maximum number of weeks beyond *maxage* that a password can be changed before administrative action is required to change the password (root is exempt).
  - *minalpha:* minimum number of alphanumeric characters the new password must contain.
  - *minother:* minimum number of non-alphanumeric characters the new password must contain. (Other characters are any ASCII printable characters that are non-alphanumeric and are not national language code points).
  - *minlen:* minimum number of characters the new password must contain.
  - *maxrepeats:* maximum number of times a character can appear in the new password.
  - *mindiff:* minimum number of characters in the new password that must be different from the characters in the old password.
  - *histexpire:* number of weeks that a user will not be able to reuse a password.
  - *histsize:* number of previous passwords that cannot be used.
  - *dictionlist:* list of dictionary files checked when a password is changed. Dictionary files contain passwords that are not allowable.
  - *pwdchecks:* list of external password restrictions methods that are used when a password is changed.

- Trusted Computing Base

  The Trusted Computing Base (TCB) is the part of the system that is responsible for enforcing the information security policies of the system. All of the computer hardware is included in the TCB, but the administrator of the system should be concerned primarily with the software components of the TCB. When you install AIX Version 4.1.x or 4.2.x, you have the option to select the installation of TCB. Selecting **yes** enables the trusted path, trusted shell and system integrity checking (`tcbck` command). Selecting **no** disables these features.

  Note

  The TCB features can only be enabled at installation time.

  The TCB software consists of:

  - The kernel (operating system)
  - The configuration files that control system operation
  - Any program that is run with the privilege or access rights to alter the kernel or the configuration files

  Most system files are accessible only by the root user; however, some can also be accessed by members of an administrative group. Only the root user can alter the operating system kernel.

- File system security

  All file system objects (including files, directories, special files, link files, symbolic link files, and pipes) have security mechanisms associated with them. The most commonly used is the access control list (ACLs), but the following additional ways of controlling file security can also be used:

  - Base Access Control Lists (ACLs)–Specifies the permissions for the owner, group, and others. These permissions are controlled through the `chmod` command.
  - Extended ACLs–Provides finer access control than the base ACLs.

- Auditing

  The auditing subsystem provides the system administrator with the means to record security-relevant information, which can be analyzed to detect potential and actual violations of the system security policy. The auditing subsystem has three functions: event detection, information collection, and information processing. These three functions can be described as follows:

  1. Event Detection

     Event detection is distributed throughout the Trusted Computing Base (TCB), both in the kernel (supervisor state code) and in the trusted programs (user state code). An auditable event is any security-relevant occurrence in the system. A security-relevant event is any change to the security state of the system, any attempted or actual violation of the system access control or accountability policies, or both.

     The programs and kernel modules that detect auditable events are responsible for reporting these events to the system audit logger, which runs as part of the kernel. To control event detection at the global level, use the `audit` command to turn on or off the audit subsystem. To control event detection at the local level, you can audit selected users for groups of audit events (audit classes).

  2. Information Collection

     Information collection includes logging the selected auditable events. This function is performed by the kernel audit logger, which provides both a System V Communication (SVC) subroutine and an intra-kernel procedure call interface that records auditable events. The audit logger appends each successive record to the kernel audit trail, which can be written in one (or both) of two modes:

BIN mode	The trail is written into alternating files, providing safety and long-term storage.
STREAM mode	The trail is written to a circular buffer that is read synchronously through an audit pseudo-device. STREAM mode offers immediate response.

  3. Information Processing

     AIX provides several options for processing the kernel audit trail. The BIN mode trail can be compressed, filtered or formatted for output. The STREAM mode audit trail can be monitored in real time to provide immediate threat monitoring capability.

3.1.1.3 SunSoft Solaris

Solaris from SunSoft uses a multidimensional approach to system and network security to protect against unwanted intrusions. Below, we cover some of these dimensions of SunSoft Solaris security.

- Controlling Login Access

  Login access control consists of a set tools that help administrators control who can log in to the system. The main feature is the use of a password which can be used to check the identity of the person who is attempting to log in. Solaris uses a set of management features including:

Password validation	Solaris compares the password the user provides to the one set by and stored for that user in a special file known as the *Shadow Password* file. If the password matches, the user is allowed to log in.
Password aging	Solaris enables the administrator to set an expiration date for passwords (a common technique in all UNIX systems).
Disallow old password	This feature prevents a user from reusing a previously used password.
Shadow password file	A hidden file (called /etc/shadow) stores all user passwords and is readable only by root.

- System resource access control

  Solaris includes the Automated Security Access Tool (ASET), which can automatically assess the state of the system and place it in one of three predetermined security states: low, medium or high. If run periodically, ASET will alert the administrator of any security breaches.

- File protection on Solaris

  Solaris implements two methods for file protection: traditional *UNIX-style* permission settings and Access Control Lists (ACLs). As described earlier in this chapter, with UNIX-style permission setting, it is possible to set read, write, and execute permission indications for a file's owner, selected groups, or the world (also known as other).

  Similar to ACLs in AIX, support for Access Control Lists (ACLs) are something new in Solaris. With ACLs, extensive lists of authorization information can be maintained for every file, enabling a finer granularity of control over file access. For example, with ACLs, access can be controlled on a per-user basis in addition to on a per-group basis.

- Auditing

  Auditing helps administrators track security related events, including many types of access attempts. Solaris has two types of auditing methods, UNIX system logs and C2 auditing.

  - UNIX system logs (syslogs) keep track of login events, resource usage, quotas, and so forth.
  - C2 auditing, also called Control Access Protection, can produce a more detailed audit report. C2 can create audit logs by user, event and class.

3.1.1.4 Hewlett Packard HP-UX 10

The core of the HP-UX 10 is the Trusted Computing Base (TCB), a set of mechanisms responsible for enforcing the system's security policies. These policies are enforced to protect information from disclosure to unauthorized individuals.

HP-UX 10 supports information labeling and sensitivity labeling in conjunction with mandatory and discretionary access control policies to mediate access to system information. The security features of the HP-UX 10 Operating System include:

- Sensitivity Labeling

Sensitivity labels are assigned to all system subjects (for instance, users and processes) and objects (such as files and devices). A system administrator or security officer is responsible for defining each user’s clearance, thereby determining the range of sensitivity labels under which a given user is permitted to operate.

- Information Labeling

  This label is another security attribute and deals with the content of an object rather than the object itself. On files, this label reflects the highest level of information contained in a file. The label automatically “floats up” as information of a higher security level is added. For processes, the information label floats up as the process reads more sensitive information.

- Mandatory Access Control (MAC)

  When users attempt to access objects, their sensitivity labels are compared to determine access rights. This procedure supports the concept of read-down, and write-equal and, with the appropriate authorization, write-up. Users can read objects at their own level or lower and write only to objects of the same level. This policy is referred to as *mandatory* since users cannot alter these access rights at their own discretion.

- Discretionary Access Control (DAC)

  This policy allows users to grant or deny access at their own discretion within the limits of MAC. HP’s Trusted OS enforces this policy through the implementation of ACLs that grant or deny any user access to objects.

3.1.2 Microsoft Windows NT Server

Windows NT’s security system is based on the following features:

- Permissions and rights
- Domain resource management model
- Users and groups and security identifiers
- System administrator and operator privileges
- Resource protection through access control lists and security descriptors
- Inter-domain communication via trust relationships
- Security data structures
- Events and auditing

Each of these areas are described briefly here. Refer to Microsoft Windows NT documentation such as the *Windows NT Resource Guide* for more information.

3.1.2.1 Permissions and Rights

Windows NT uses the terms *permissions* to refer to what access a user has to resources (see Section 3.1.2.5, “Access Control Lists and Security Descriptors” on page 26). Rights (or User Rights) are defined as what tasks a user is permitted to do. For example, a user may have read, write and execute ***permissions*** for a file’s resource and may have the ***rights*** to log on to a server locally or shut down the server.

3.1.2.2 Domain Resource Management Model

As well as features protecting resources on individual machines, Windows NT uses the concept of a domain to define a set of resources spanning one or more servers. The domain model centralizes administration and security for all servers identified as members of the domain. This provides:

- User account validation against a common user database known as the Security Account Manager (SAM) database
- Assignment of resource permissions to users defined in the SAM database
- Access to all resources in a domain from a single user account
- Administration of users by clustering them in groups

Windows NT defines one server as the Primary Domain Controller (PDC) where the SAM database is maintained. Other servers operate in the domain as Backup Domain Controllers and they maintain a copy of the SAM database. A Windows NT server can also operate independently of a domain in the same network. This server needs to be managed separately, but can still share its resources with other users in the domain/network if required.

3.1.2.3 Users, Groups and Security Identifiers

A Windows NT user account consists of a user name and password, the groups in which the user has membership and any permissions or rights the user has defined for using the system. While a user account record can be manipulated directly to set permissions and rights for a user, the normal model for administering users' permissions and rights is through groups–see below. In the domain model, the PDC maintains this account information for the collection of servers in the domain. Windows NT, by default, has two built-in accounts, Administrator and Guest. (TME 10 Security Manager adds the user TMERSRVD.)

Windows NT domain user logons can be restricted to certain hours of the day and/or days of the week or to named workstations.

Windows NT implements a range of user and password control functions:

- Set password expiry periods and/or user account expiry date
- Set password rules such as minimum and maximum password lengths
- Lock out the user after a set number of failed logon attempts

A group is an account that contains user accounts and other groups as required. Access to resources can be granted to individual users, but large numbers of users can be treated as one account by having them specified in a group. Additionally, rights to perform system tasks can be granted to users or groups. The usual process is to associate a collection of access rights to a group and then make users who require those rights members of that group. For example, a user can perform system backups if their user ID is added to the *Backup Operators* group. If a user needs multiple sets of rights that would not apply to all other members of the user's group, that user can be specified in more than one group and the rights could then be specified to the alternate groups. Windows NT allows a distinction between groups defined globally to the domain and locally to individual servers.

Local Group Defined at each Windows NT server whether in a domain or stand-alone. Can contain other groups, including global groups to allow domain-wide administration. For example, by default, each local Administrators group, contains the Domain Admins global group. Adding a user to the Domain Admins group, will enable them as an administrator on any server in the domain.

Global Group Defined on PDCs (and therefore replicated to BDCs). Can contain any user defined to the domain (but not any other groups). Global groups can be given access to resources that reside on any PDC or Backup DC (BDC).

The Windows NT security model makes extensive use of a Security Identifier (SID) to uniquely identify a user or group. SIDs are defined by the system to be unique using a combination of the user information, time and date of creation and domain information.

3.1.2.4 System Administrator and Operator Privileges

The Windows NT Administrator is a form of root or superuser. A user added to the Administrator group of a server or Domain Admins global group can perform all functions on the server or servers in the domain. In order to distribute administrative functions without granting excessive powers to users, Windows NT has a number of default groups that define operator privileges. These groups provide the necessary rights to perform certain functions such as managing print queues or performing backups. As mentioned above, the usual way to manage these rights is to assign the user to a group that has the necessary rights defined, but if required, these rights can be assigned to individual user accounts.

3.1.2.5 Access Control Lists and Security Descriptors

Access *rights* determine the actions Windows NT users can perform with the system; access *permissions* determine what capabilities the user has on a particular object in the system, such as a file or printer resource. Permissions are stored in WIndows NT Security Descriptors. These are data structures that contain the security information associated with an object or resource. For each resource, the Security Descriptor contains a SID of an owner, a group association and two types of Access Control List (ACL):

Discretionary ACL	Controlled by the object owner and identifies who may and may not access the object and the type of access (Read, Update and so forth.)
System ACL	Controlled by a security administrator and controls audit message generation.

In general, the administrator need not be aware of the physical implementation of these structures. They are managed by the system automatically based on higher-level administrative operations.

Note

Much of the granular security of files resources is implemented in Windows NT through the NT File System (NTFS) file system. Capabilities, such as certain access restrictions on resources when a user is logged on locally and file-by-file ACLs, are NOT available on disk partitions where another file system

Under Windows NT, user and group permissions are cumulative. Note also that deny access takes precedence over grant access. When Windows NT checks permissions, it does so in one pass, not discriminating between users and groups. As soon as a “deny access” permission is reached, the search is terminated and access to the resource is denied.

3.1.2.6 Inter-Domain Communication through Trust Relationships

A domain provides a convenient way of dividing administrative responsibility for collections of servers and resources into logical groups. However, in an environment where multiple domains exist, there is likely to be the need for users on one domain to access resources controlled in another domain. Without some mechanism for allowing cross-domain access, each user would have to maintain a user ID and password for each domain, and that user profile will need to be maintained by administrators on each domain. In the future, Windows NT will implement some form of resource name directory to span multiple domains. The current solution is to establish *trust relationships* between domains.

A trust relationship is a link between two domains, where one domain can be configured to honor the users of another domain. The user authentication is done at the user's home domain, and when the user requests to access a resource on the trusting domain, the trusting domain only needs to confirm that the user has been authenticated by a trusted domain to allow the user access.

Trust relationships can be set up for one-way or two-way operation. In a one-way trust, users from one domain (the trusted domain) can access resources defined on a second (trusting) domain but users on the trusting domain cannot access resources on the trusted domain. Two-way operation, as its name suggests, allows resources on each domain to be shared among users from the other domain. Microsoft outlines four domain models to manage the users and resources for different-sized networks:

Single	All users and resources are defined in one domain. No trusts are necessary.
Master	One domain is determined to be the Master containing all user and group records. Other domains trust this domain to allow Master Domain users access to their resources.
Multiple Master	Resource domains trust two or more Master Domains. The account information can be divided between the master domains. The Master Domains can trust one another to allow full administration.
Complete Trust	Every domain has a two-way trust relationship with every other domain. Each domain maintains its own set of users, but through the complete trusts, users can access resources on all domains. The trust links get very difficult to maintain as the number of domains grows. The number of trusts required is n*(n-1), where n is the number of domains (for instance, four domains require 12 trust links; six domains require 30!).

3.1.2.7 Alerts and Auditing

Windows NT includes the capability of auditing the use of resources, providing a log of such events as failed logons, attempted unauthorized access to files and so on. Audit thresholds can be established to generate alerts. Other alerts can be signalled by Windows NT, such as power loss from an uninterruptable power supply. User or group IDs can be individually registered to receive the alerts generated by the system.

3.2 Critical Operating System Resources

The purpose of this section is to identify some of the most important operating system resources of each platform that need to be protected. When you implement TME 10 Security Management, you need to remember that many application-specific resources and vital datasets will have to be protected in addition to the operating system resources mentioned here.

3.2.1 UNIX Platforms

On all UNIX operating system flavors, there are a number of resources that must be securely protected at all times. These resources not only deal with user ID, group ID, password, or permission but also with network access control files, which are very tempting to violate by external, internal or even trusted personnel that act as intruders to the system.

These resources and operating system flavors are summarized in the following table:

Table 2. Different UNIX Protected Resources

Operating System	Resource	Permissions	UID	GID
Generic UNIX	/etc/passwd	-rw-r--r--	root	system
	/etc/group	-rw-r--r--	root	security
	/etc/hosts.equiv	-rw-r--r--	root	system
	/$HOME/.rhosts	-rw-r--r--	root	system
	/etc/exports	-rw-r--r--	root	system
AIX/6000	all generic UNIX resources, plus:			
	/etc/security/passwd	-rw-r--r--	root	security
	/etc/security/group	-rw-r--r--	root	security
	/etc/security/user	-rw-r--r--	root	security
SunSoft Solaris	all generic UNIX resources, plus:			
	/etc/ttytab	-rw-r--r--	root	system
	/etc/shadow	-rw-r--r--	root	system
	/etc/default/login	-rw-r--r--	root	system
HP-UX 10	all generic UNIX resources			

3.2.1.1 The .rhosts File

The .rhosts file is similar in concept and format to the hosts.equiv file, but allows trusted access only to specific host-user combinations, rather than to hosts in general. Each user may create a .rhosts file in his/her home directory and allow access to their account without a password. Most people used this mechanism to allow trusted access between accounts they have on systems owned by different organizations who do not trust each other’s hosts in hosts.equiv.

This file presents a major security problem: While hosts.equiv is under the system administrator’s control and can be managed more effectively, any user may create an .rhosts file granting access to whomever he/she chooses, without the systems administrator’s knowledge.

As a rule of thumb, the best way to manage .rhosts files is to completely disallow them on the system.

3.2.1.2 The exports File

The /etc/exports is one the most important parts of Network File System (NFS) configuration. This file lists which file systems are exported (made available for mounting) to other systems. This file presents a major security problem since anyone who can mount your file system via NFS can then use it at their leisure.

We must avoid granting “root” access to a host on an NFS file system since a host that has root access to a file system has compete control to the file system. Untrusted hosts should never be given root access to any NFS file systems.

3.2.2 Microsoft Windows NT Server

By default, the Windows NT resources that need to be protected are only susceptible to alteration by administrators or users assigned certain administrative rights. The Windows NT system files that require protection are not, by default, available to users logging on from workstations on the network.

Note

As with any system, it is very important in a Windows NT environment to ensure the physical location of the file server is secure. Even though files can be protected against users attempting to logon locally, other malicious damage can be inflicted, for example by rebooting the machine from a floppy disk.

While administrative functions are protected against ordinary users logged on locally, a user who can log on at the server keyboard will have access to much of the local file system as well as some keys in the NT Registry. General users should not be given the right to log on locally at a server. Note that members of the Administrators group and the operator groups (Account, Backup, Print, and Server operators) are all given local logon capability.

Avoid compromising the security of the Windows NT system root directory, usually named something like C:\WINNT. This includes the Security Account Manager database and other system areas such as the Registry.

Windows NT Version 3.51 allows users access to remote registries through REGEDT32.EXE by default. In Version 4.0, this has been restricted to administrators.

Chapter 4. Security in the TME 10 Environment

This chapter describes the security-related functions that are built in to the TME 10 Environment. First, the physical implementation of security is covered (maintenance of data integrity, the use of encryption and so on); then we look at the implementation of security concepts such as authorization roles. Finally, we cover some topics on the administration of user IDs in the TME 10 environment.

The topics discussed here are those areas of the TME 10 environment that are directly relevant to security issues. Refer to the TME 10 manuals or documents such as SG24-4948, *Understanding Tivoli TME 3.0 and TME 10*, for coverage of more general TME 10 environment topics.

4.1 TME 10 Framework Recap

In the era of distributed computing and multivendor environments, diverse systems are networked throughout the world. Unfortunately, the diversity that lets users choose the system that best meets their needs also creates an administrative nightmare. It requires system administrators to use a different management scheme for each hardware platform linked to the network. To do so, they must invest a considerable amount of time and money to gain experience and expertise on all the multiple flavors of administrative approaches.

As a result of this situation, various organizations, such as IBM Corporation, Hewlett-Packard Company, Tivoli Systems, Legato Systems, and others, decided to work on this subject. We focus our attention on Tivoli's WizDOM. This is an object-oriented framework with a set of services that governs object interaction. These provide a cohesive model of management, along with graphical display services and a command-line interface, which is also the cornerstone of the TME 10 Framework implementation of today.

A framework consists of software that provides a layer of isolation between applications above it and specific implementations of resources or services beneath it (Refer to Figure 5).

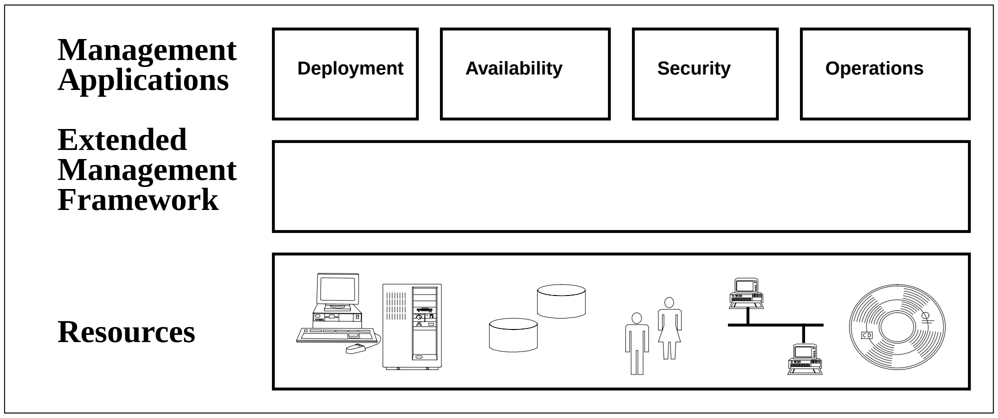


Figure 5. Framework Architecture

Tivoli's object-oriented framework provides a platform-independent architecture and includes a set of managers, brokers, and agents based on the Object Management Group/Common Object Request Broker Architecture (OMG/CORBA) specifications.

Information

The Object Management Group's CORBA specification is a widely endorsed architecture for a distributed object environment. In this architecture, objects are autonomous and potentially long-lived. Their state can be stored persistently so that when re-activated, they can start at a previously preserved point.

An open, object-oriented framework is created to provide an infrastructure with flexible interfaces and a set of applications. This helps us simplify the management of distributed systems. By distributed systems, we mean a set of interconnected systems which place mutual trust in certain security services that are shared or are common to the systems, such as authentication and access control.

This software layer (framework) is a set of prefabricated building blocks that programmers can use, extend, or customize for specific computing solutions. It allows them to develop applications without the necessity of starting from scratch since frameworks are built from collections of objects; so both the design and code of a framework can be reused. One major goal of an object-oriented framework in distributed-systems management environments is to provide a secure platform for distributed applications that makes the access control and authentication mechanisms transparent to the application level. That means, for example, that an application can work with a collection of systems or resources, with an assumption that the framework has verified the authenticity or correct accesses required.

A systems management environment based on objects does not necessarily replace management mechanisms present in the operating system. Instead, it provides easy-to-use abstractions (objects) that hide the messy details and complex management tasks.

Tivoli's primary focus was to provide a management platform that would be able to simplify day-to-day management of hundreds or thousands of computers, mainly desktop machines, in a heterogeneous environment. This was, of course, a huge task, but one that has now been accomplished using state-of-the-art, object-oriented client server technology. Security management was not initially a primary issue. The TME 10 Framework implemented uses authentication, authorization, encryption, and data integrity mechanisms, and Tivoli is making further improvements to the TME 10 Framework with full-packet encryption, authentication between TME 10 machines using session keys, MD5 hashing and digital signatures to ensure data integrity.

In the following sections, we discuss the security features implemented in the current version (3.1) of the TME 10 Framework and mention some of the new features to be included in future versions. In Appendix A, "General Security Concepts and Terminology" on page 189, we provide a general description of security-related terms and concepts, and Appendix B, "Introduction to TME 10

Framework Version 3.2" on page 195, explains a little about LCF, the new client management option in TME 10.

4.2 Network-Level Security of the TME 10 Framework

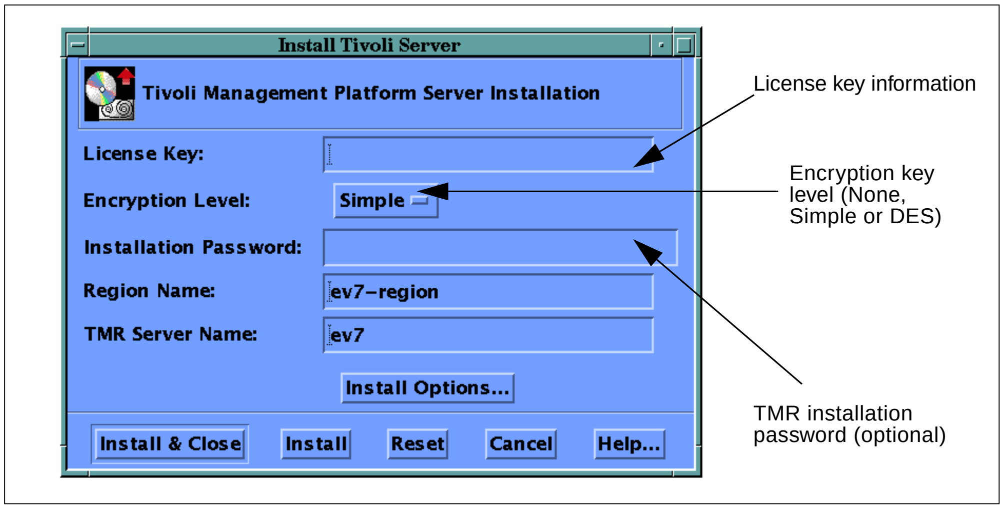
To describe the security mechanisms of the TME 10 Framework, we will break it down into two areas, the first of which is the security mechanisms in place that allow the product to operate securely on a network. Sometimes referred to as *physical* or *network-level* security, this covers areas such as how participating systems connect to each other, verify trustworthiness and so on. The second area is how the product deals with the abstract representations of the resources it manages. Resource-level security is discussed in the next section and covers items like Users, administrators and TME 10 Policy Regions.

4.2.1 Security Options and TME 10 Framework Installation

As we all know, in the TME 10 Framework environment, we must first have a Tivoli Management Region (TMR) server before we can then start adding endpoint (clients).

During the installation of the TMR server, which can take place from CD ROM or from an installation directory (file system), we are prompted for a license key and a TMR installation password (optional, by default no password is used). If the TMR installation password is used, we must provide the same password every time we subsequently install a client. This might be considered a first defense against a security breach of someone adding an unauthorized client to the management network.

The TMR installation password is used for controlling who has the ability to install TME 10 and TME 10-based applications. You can use any mix of encryption levels and encryption passwords for TME 10 intraregion, interregion and intrainstallation operations.


Install Tivoli Server

Tivoli Management Platform Server Installation

License Key:

Encryption Level: Simple

Installation Password:

Region Name: ev7-region

TMR Server Name: ev7

If you select an encryption level *Simple* or *DES*, you must also provide an encryption password. The encryption password is used during the encryption process.

Attention

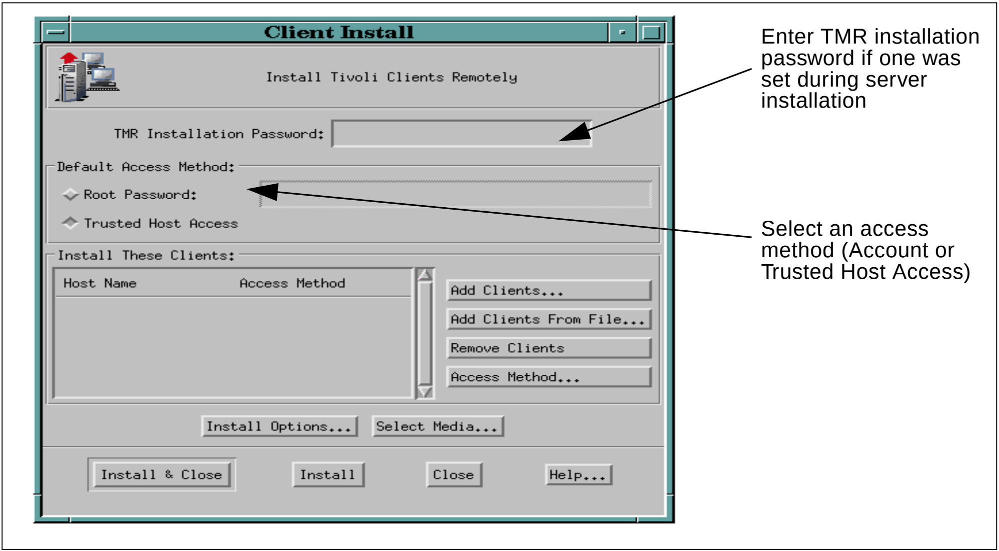
Protect your encryption password as you would protect any password; a compromised password causes a risk of a security breach.

When installing a client, we are prompted to select one of the following access methods:

- Account (UNIX rexec): The TMR server must provide the root user ID and password of the client to interact with the client. Since the *rexec* does not use the trusted host mechanism, it can be used from any host on the network. In addition to this consideration, it requires that the password be transmitted over the network, and it is susceptible to the same password snooping as unprotected services such as Telnet.

  A big flaw in rexec is that it provides error messages such as `Login incorrect` or `Password incorrect`, which a cracker can use to break into the system for valid accounts.

- Trusted Host Access: The TMR server can access the client without providing a root password since the name of the server is included in the client's /.rhosts file. Trusted access must be used in a very secure environment, and it is the system administrator's responsibility to grant or disallow the use of it.



During installation and configuration of an endpoint(client), a lifetime key and an object dispatcher number are assigned when the endpoint's object dispatcher (oserv daemon) reports itself for the first time to the TMR server. This lifetime key is used as the secret key part of each encryption-level algorithm used (the simple-XOR and DES). This key and object dispatcher number assignment is a way to secure the communication between the server's object dispatcher and the end-point's object dispatcher.

The communication between the server and endpoint object dispatchers is accomplished through Tivoli's proprietary, high-level remote procedure call (RPC) mechanism, spawned by the oserv daemons on both ends. The underlying implementation of this high-level communication protocol uses common TCP/IP port type connection between object dispatchers (port 94 is used by Tivoli).

A TCP/IP port provides a reliable, ordered, two-way transmission stream between two programs running on the same or different computers. *Reliable* means that every byte transmitted is guaranteed to reach its destination and that each byte arrives in the order in which it was sent.

Compare this with a UDP/IP port, which provides basically the same functionality of a TCP/IP port. Packets can be sent from a port on a host to a port on another host, but this is considered an unreliable system since there is no guarantee that every packet sent will be delivered or that packets will be delivered in order.

Note

During file distribution, the distribution process runs through the oserv daemon and through a socket connection between the two methods executing the distribution.

4.2.2 Authentication

Authentication verifies the identity of a *principal*. A principal can be someone (user) or something (program).

The TME 10 administrator is also called the TME principal. When a TME principal is created, you must define all the system user IDs that will be mapped to that TME principal. Those IDs will get that principal's TME 10 desktop. See Section 4.3.2.2, “TME 10 Administrator Logins” on page 48, for more on this subject. The mapping occurs without any further authentication checks; the user ID is considered authenticated to TME 10 when the user passes the operating system authentication.

TME 10 supports two authentication mechanisms: the standard UNIX password verification to authenticate TME 10 administrators and the other one is Kerberos authentication.

TME 10 software provides a Kerberos Network Authentication Service Version 4 implementation that can be used to authenticate the TME 10 administrator and to replace the native operating system authentication. Authentication still occurs at the operating system level, only now using Kerberos.

Kerberos is used in TME as an additional step to secure the computer network. Password-based authentication is not too secure since passwords sent across a network can be intercepted and subsequently used by eavesdroppers to impersonate the user. Kerberos is a distributed authentication service based on cryptography, which allows a process (a client) running on behalf of a principal (a user) to prove its identity to a verifier (a TMR server) without sending data across the network that might allow an attacker to impersonate the principal (see Section 4.2.5, “Kerberos” on page 37, and Appendix A.4, “Kerberos” on page 191, for more details).

4.2.3 Authorization

Authorization verifies that the principal has sufficient rights to perform an operation. Each activity in TME 10 has an associated set of roles that allow invokers access to the activity. For any operation on an object, the authorization process involves checking to see if the principal has at least one of the roles required by the activity. If so, the attempt to run the operation is permitted.

Therefore, objects have an Access Control List (ACL), which basically states who may access the object. If a user has been previously authenticated (is a TME principal) and tries to access an object, the ACL on the object must allow access to the principal’s role; then the activity is allowed to perform the specific action on the object.

4.2.4 Encryption Levels on TME 10 Framework

TME 10 provides a user-selectable encryption facility to protect security credentials between Object Request Brokers (ORBs). Security credentials include sensitive data such as inter-region passwords, TME 10 administrator logins and roles, as well as authentication requests and authentication data. TME 10 provides three levels of encryption for ORB security credentials:

1. **None**

   TME 10 does not encrypt any of the TME 10 security data except for key distribution. Unless you have a very secure network and all of the systems on your network are trustworthy, it is not recommended that you use this level of encryption. There is no encryption, only when keys are distributed.

2. **Simple**

   This level of encryption provides a simple, key-based encryption that protects TME 10 security credentials from casual viewing. The impact of this encryption mechanism on performance is minimal. We must keep in mind that this encryption scheme is not unbreakable, but time and effort is required to crack the encryption and snoop the data.

   This simple method of encryption uses the mathematical function known as exclusive OR (XOR). Where if you XOR a number with a second number, you obtain a third number. If you XOR the third number with the second number again, you get back your starting number.

   When using this encryption-level, TME 10 encrypts everything (for example, method arguments and out-of-band interobject messages IOM data), with the exception of application data.

3. **DES**

   When used for communication purposes, such as in a TME 10 environment, both sender and receiver must know the same secret key, which is used by both to encrypt and decrypt the message.

   TME 10 includes an implementation of DES (Data Encryption Standard) for use in those environments where a high degree of security and authentication control is required. Keep in mind that when using DES there is an impact on performance, estimated at around 12 percent or more of communication overhead.

   TME 10 encrypts everything, and application data is encrypted using DES.

4.2.5 Kerberos

TME 10 provides an implementation of the Kerberos network authentication service Version 4 from Massachusetts Institute of Technology (MIT). Kerberos provides an authentication mechanism for TME 10 administrators.

Kerberos uses DES cryptography to pass sensitive data (such as passwords, security credentials, TME 10 logins, and so forth) around an open network and also to decrease the possibility for a user or service to masquerade as another user or service (see Appendix A.4, “Kerberos” on page 191, for more details).

During TME 10 Framework installation, TME 10 uses, by default, standard UNIX passwords to authenticate TME 10 administrators. However, TME 10 enables you to configure your system so that Kerberos is used as the means by which a TME 10 administrator (TME principal) is authenticated.

TME 10 sets the logins, Kerberos principal names, and managed nodes that are valid for an administrator. Administrators can have different logins for different managed nodes.

When defining an administrator as being valid from a specific managed node (for example, to enable an administrator to start his or her desktop from a specific machine), the administrator must have a valid Kerberos principal on the managed node.

Logging into Kerberos

Assuming TME 10 is configured to allow Kerberos authentication, the following steps are used to start the desktop:

1. Log in as yourself on one of the machines for which you have a Kerberos user principal defined.
2. Ensure that you TME 10 environment is set correctly by running:

   `. /etc/Tivoli/setup_env.sh` (Bourne shell)

   -OR-

   `. /etc/Tivoli/setup_env.csh` (C shell)

3. Log in to Kerberos by running the `kinit` command:

   ```
   $ kinit
   Principal name: user-principal-name
   Instance: instance-name
   Password:
   ```

   where `user-principal-name` is the principal name assigned to your TME 10 administrator for this machine, and `instance-name` is the name associated with your principal name.

4. Enter the password assigned to you.
5. Run the `tivoli` command to initiate your TME 10 desktop.

Note

TME 10 checks the lifetime of your Kerberos ticket every five minutes. You can set the ticket expiration time using the `kbd_edit` command. If your Kerberos ticket expires while your desktop is running, you see an error message that states:

```
Your Kerberos ticket has expired.
 Re-run kinit to retrieve another one.
```

Use the `kinit` command to reestablish your Kerberos credentials. You do not have to log out of TME 10 for these new credentials to become effective.

4.2.6 Data Integrity

TME uses some data integrity mechanisms to ensure the confidentiality and security of its information. For example, an MD5 hash function is used during file distribution transmission. Also, all tasks are entirely encrypted during their distribution.

TME 10 Framework Version 3.2 (also known as Lightweight Client Framework or LCF) introduces MD5 (message digest) hashing on the message header and the signing of packets. This enhancement has been added due to the fact that LCF accommodates a far larger number of systems in a TMR. Changes were needed to secure the communication between LCF gateways and their endpoints.

This concept of hashing is used in TME 10 when a message is transmitted with a MAC (message authentication code). A message authentication code (MAC) is a one-way, hash-computed secret key equivalent to a public key signature derived from a message and some secret data. Its main purpose is to detect if the message has been altered.

The message authentication code (MAC), employs an encryption routine for a data stream and from this routine develops a unique code that is appended to the data stream. It is important to note that only the appended code is actually encrypted, and the message remains in plain text. The process, repeated by the recipient, must then produce the identical code before the data can be accepted as valid or unaltered.

An MD5 hash is a 128-bit quantity (sometimes called a checksum). The TME 10 Framework 3.2 uses the first 64 bits for the MAC (message authentication code).

4.2.7 TMR Connections

TME 10 provides the ability to subdivide an enterprise network into multiple Tivoli Management Regions (TMRs) and then to connect them with one-way or two-way connections (so-called interregion connections).

Multiple TMRs are needed in situations where a collection of clients cannot be contained within a single TMR or where organizational or management requirements make it essential. The 3.1 of the TME 10 Framework supports a recommended maximum of 200 clients. Therefore, if the network contains more than 200 clients, you would create numerous TMRs and connect them.

A factor that affects the number of TMRs needed to manage an enterprise is the need to perform local administration. If your desire is to achieve centralized system administration—that is, administrators in a single location administering distributed systems—a single TMR will be your best choice. However, if you use localized system administration (administrators at various operational sites), multiple TMRs may be more appropriate.

Note

Using multiple TMRs is not an automatic method of providing system redundancy for failing system management servers.

There are two security considerations that might mean selecting a configuration of multiple TMRs:

1. Encryption: A TME 10 installation cannot mix encryption levels within a single TMR. All nodes within a TMR must use the same encryption level (None, Simple or DES). If you need different encryption methods between clients, you must create multiple TMRs.
2. Limited access: Restriction of what local administrators can do on machines elsewhere in the enterprise.

   An example of limited access is to give a local administrator the ability to perform administration tasks in a particular area but more limited or different capability for all other machines. One way to accomplish this goal is to create multiple TMRs and use a one-way interconnection between the TMRs.

To connect two TMRs, you must perform the connection from either one or both TMRs, and you must have the super administration role for both TMRs to accomplish the connection.

While connecting two TMRs, each TMR must supply some information about themselves, such as:

- The TMR server name
- Region number
- The encryption level and password in use in the other TMR

This information is used when the remote TMR is registered in the local TMR.

As mentioned above, you could connect two TMRs with different encryption levels, but such a configuration should be implemented with caution because you could compromise the security of a TMR.

Connecting TMRs through Secure vs. Remote Connection

Connecting one TMR to another through a secure connection does not require a remote login to set up the connection. Therefore, the root or administrator password would not be sent across the network, and the opening up of remote logins (using */.rhosts*, */etc/hosts.equiv* or their equivalent) is not necessary. Connection requests are required by each communicating TMR. Each side must do the following:

- Add the remote TMR’s host name, region number, and encryption level to the interregion object (`wlookup` InterRegion).
- Try to communicate with the remote TMR.

- If communication is successful, add the TMR name to the interregion object and exchange resources.

A remote connection allows one TMR to provide all of the connection request information and make the connection. Remote connections perform the same steps as a secure connection, but the `rexec` and `rcmd` is used to communicate with and start the connection to the remote TMR. Caution must be exercised when using this type of connection since the TMR establishing the connection must have the remote TMR server's trusted hosts or root password.

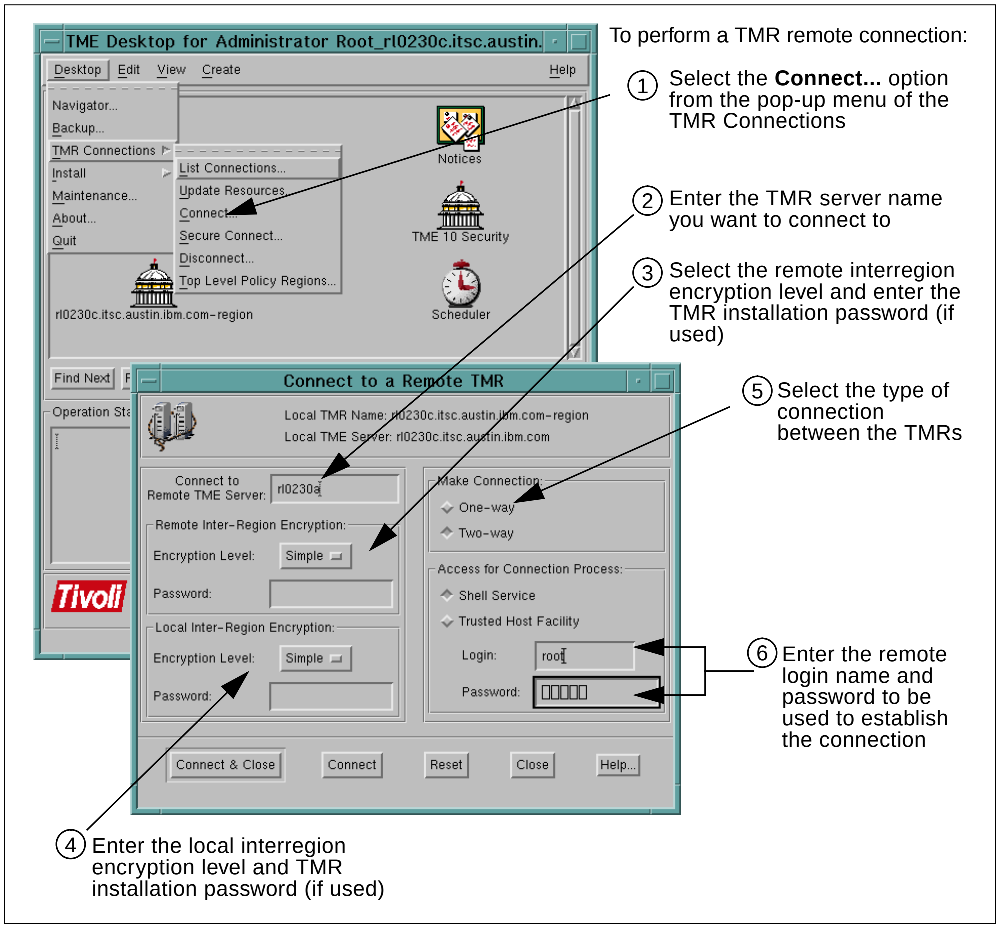
One-way vs. Two-way Connections
One- and two-way connections determine the visibility of interconnected TMR's resources.

A one-way connection enables one of the TMRs to act as the resource manager of the other TMR's resources. The managed TMR will not be able to view the managing TMR's resources.

A two-way connection allows interconnected TMRs to view all of the other's resources, as long as they are updated regularly—a process that is not automatic by default. An administrator can configure a scheduled job to perform this update function.

The following screen dialogs illustrate what to expect when making TMR connections. For our examples, we use two-way remote and secure type connections.

Remote connection example



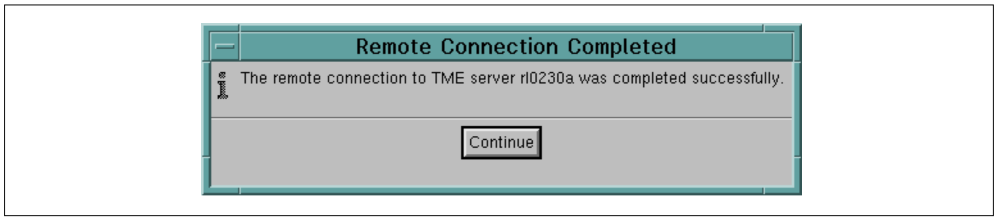
Local TMR Name: rl0230c.itsc.austin.ibm.com-region
Local TME Server: rl0230c.itsc.austin.ibm.com
Connect to Remote TME Server: rl0230a
Login: root

To perform a TMR remote connection:

1. Select the **Connect...** option from the pop-up menu of the TMR Connections
2. Enter the TMR server name you want to connect to
3. Select the remote interregion encryption level and enter the TMR installation password (if used)
4. Enter the local interregion encryption level and TMR installation password (if used)
5. Select the type of connection between the TMRs
6. Enter the remote login name and password to be used to establish the connection

Figure 6. TMR Remote Connection

If the TMR remote connection succeeds without any errors, you receive an informative screen like this:



The remote connection to TME server rl0230a was completed successfully.

Figure 7. Remote Connection Completed Successfully

Secure connection example

When making a TMR secure connection, you must know ahead of time the TMR region number of the remote TMR server you wish to connect to. You could find this information by running the `odadmin odlist` command at both ends, as shown in the example.

```
$ . /etc/Tivoli/setup_env.sh (csh)
```

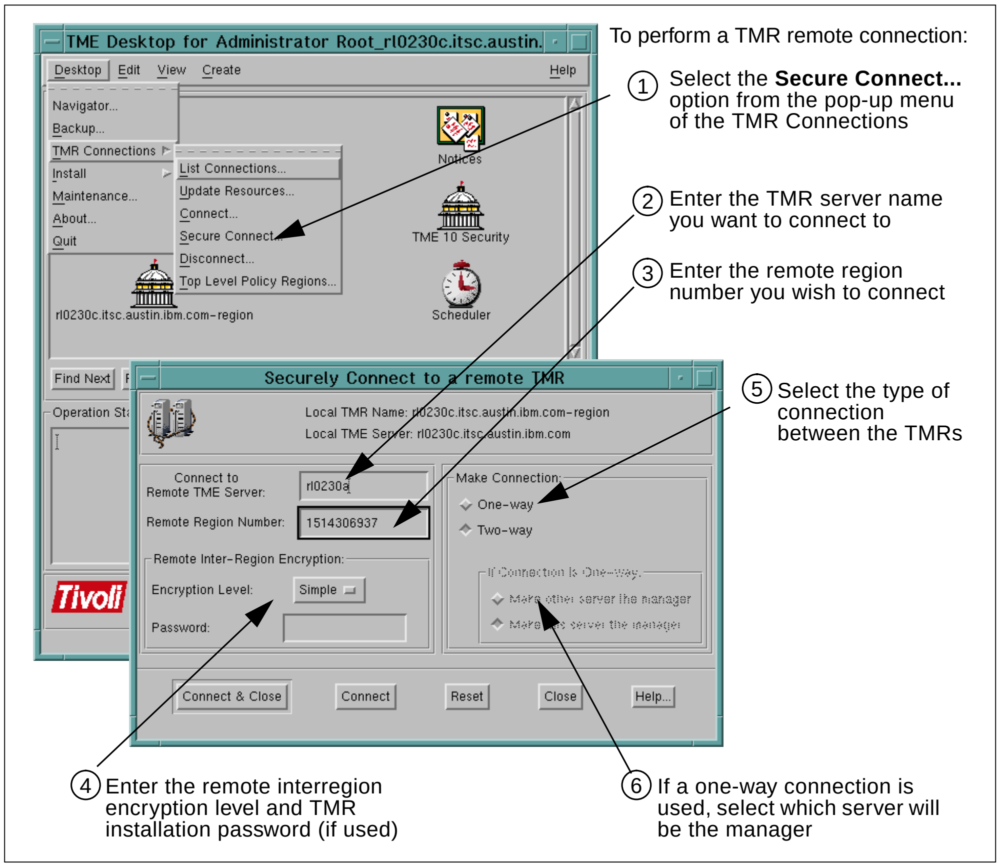
Run this command to set up your TME 10 working environment properly

```
$ odadmin odlist
```

Execute this command at both ends to find out the region numbers you want to connect

```
Region      Disp  Flags  Port         IPaddr   Hostname(s)
1514306937     1    ct-    94      9.3.1.113   rl0230a.itsc.austin.ibm.com
```

Region number you must use for the secure connection



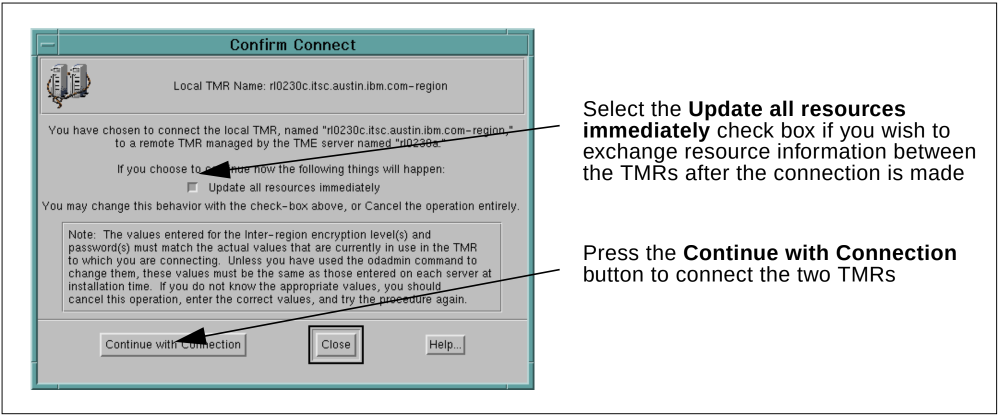
Local TMR Name: rl0230c.itsc.austin.ibm.com-region
Local TME Server: rl0230c.itsc.austin.ibm.com

Connect to Remote TME Server: rl0230a
Remote Region Number: 1514306937
Remote Inter-Region Encryption:
Encryption Level: Simple
Password:
Make Connection:

To perform a TMR remote connection:

1. Select the **Secure Connect...** option from the pop-up menu of the TMR Connections
2. Enter the TMR server name you want to connect to
3. Enter the remote region number you wish to connect
4. Enter the remote interregion encryption level and TMR installation password (if used)
5. Select the type of connection between the TMRs
6. If a one-way connection is used, select which server will be the manager

Figure 8. TMR Secure Connection


Confirm Connect

Local TMR Name: rl0230c.itsc.austin.ibm.com-region

You have chosen to connect the local TMR, named "rl0230c.itsc.austin.ibm.com-region," to a remote TMR managed by the TME server named "rl0230a."

If you choose to continue now the following things will happen:

☐ Update all resources immediately

You may change this behavior with the check-box above, or Cancel the operation entirely.

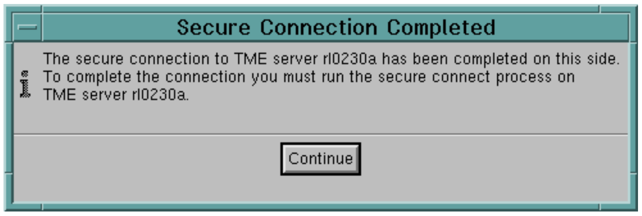
Note: The values entered for the Inter-region encryption level(s) and password(s) must match the actual values that are currently in use in the TMR to which you are connecting. Unless you have used the odadmin command to change them, these values must be the same as those entered on each server at installation time. If you do not know the appropriate values, you should cancel this operation, enter the correct values, and try the procedure again.

Select the **Update all resources immediately** check box if you wish to exchange resource information between the TMRs after the connection is made

Press the **Continue with Connection** button to connect the two TMRs

Figure 9. Confirm Connection Dialog for a TMR Secure or Remote Connection

The dialog in Figure 9 is common for remote or secure TMR connections. When TMRs are first connected, the administrator is asked if resources should be immediately updated. If the connection is a two-way connection as in our case, resources are exchanged immediately between the two TMRs. For a one-way connection, if you complete the connection on the managing side, then the resource exchange occurs immediately. If you complete the connection on the managed side, you must update the resources from the desktop or manually using the `wupdate -u` command. Some process should be put in place to ensure the continued regular updating of interregion TMR information.


Secure Connection Completed

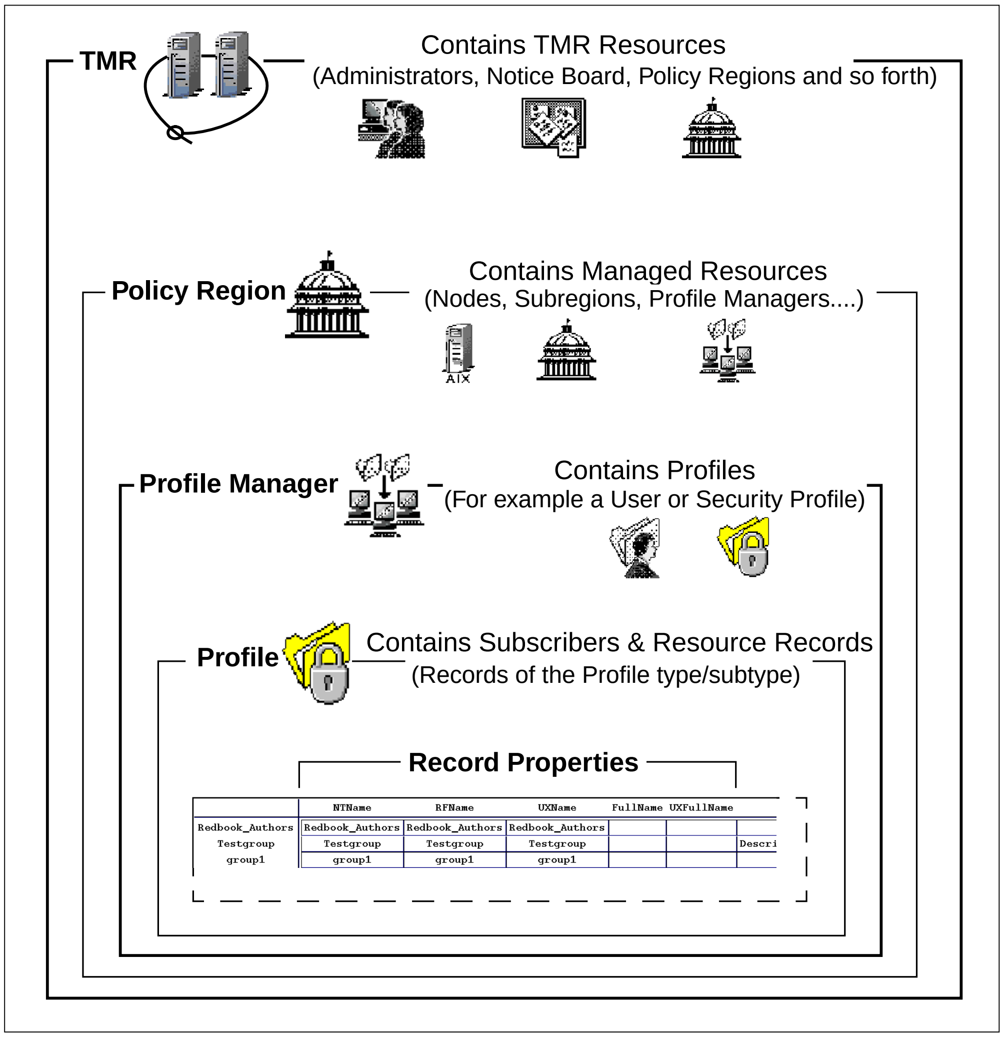
The secure connection to TME server rl0230a has been completed on this side. To complete the connection you must run the secure connect process on TME server rl0230a.

Figure 10. Secure Connection Completed Successfully

As you can see in the dialog in Figure 10, once you complete the TMR secure connection on one TMR server, you must execute the same process at the other to properly establish the connection.

4.3 Resource-Level Security of the TME 10 Framework

The TME 10 Framework contains the necessary security features to ensure system and network resource management is limited to those individuals with the correct responsibilities. This is implemented through means such as TME 10 specific administrator types, roles relating to the management of the TME 10 Management Regions (TMRs) and resources, and the use of policies in policy regions, each of which are discussed here. The following picture demonstrates the hierarchy of TME 10 resource management, especially for management applications like TME 10 Security Management that use profile-based management. You may find it useful to refer to this picture while reading the sections that follow.


TMR
Contains TMR Resources
(Administrators, Notice Board, Policy Regions and so forth)
Policy Region
Contains Managed Resources
(Nodes, Subregions, Profile Managers....)
AIX
Profile Manager
Contains Profiles
(For example a User or Security Profile)
Profile
Contains Subscribers & Resource Records
(Records of the Profile type/subtype)
Record Properties

	NTName	RFName	UXName	FullName	UXFullName	
Redbook_Authors	Redbook_Authors	Redbook_Authors	Redbook_Authors			
Testgroup	Testgroup	Testgroup	Testgroup			Descri
group1	group1	group1	group1			

Figure 11. The TME 10 Resource Management Hierarchy

4.3.1 TME 10 Management Regions

The scope of system and network resource management in the TME 10 environment is known as a TME 10 Management Region (TMR - often written as Tivoli Management Region). A TMR consists of a TME 10 management server and any clients managed under that server. Depending on your organization, you may manage all resources under a single TMR, you may have separate parts of the organization managed under different TMRs or have some combination of TMRs with management shared through links between them. The *TME 10 Framework Planning and Installation Guide* details recommendations for the deployment of TMRs.

Each TMR maintains a *name registry,* which is a table containing location information of every resource type managed in a TMR. For most resources in TME 10, their registration in the name registry is handled automatically by the TME 10 application in use.

4.3.1.1 TMR Connection

An administrator with the required role (super) in two TMRs can connect the TMRs together for management purposes.

A second TMR can be connected remotely through either the trusted host facility or remote login access. If you do not wish to send UNIX root or NT Administrator passwords across the network or open up remote logins, then you can use a *secure connect* facility. Secure connect requires each half of the connection to be set up locally at each TMR server (see Section 4.2.7, “TMR Connections” on page 38, for more details).

Once the connection is defined at each server, resource information is shared between the servers using the TME 10 Framework-configured level of encryption. A connection between two TMRs can be made one-way or two-way.

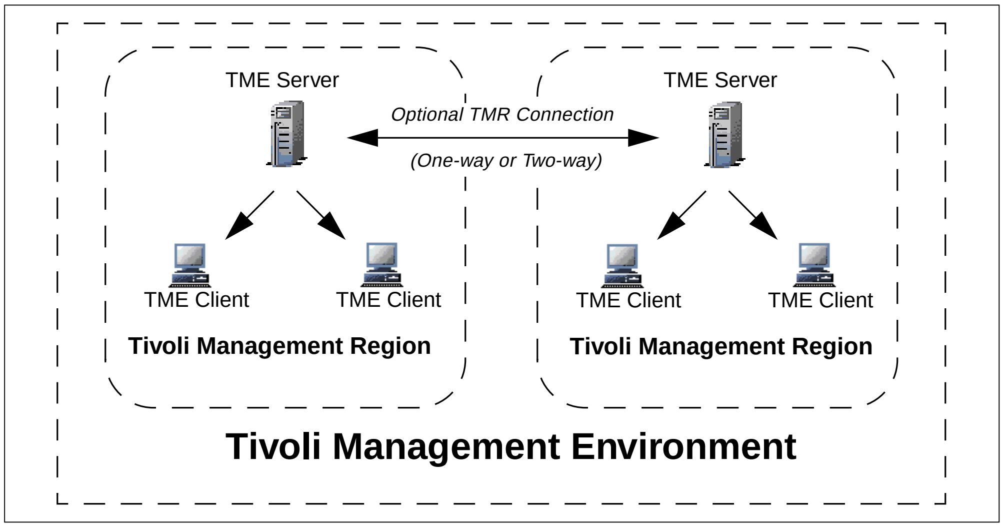


Figure 12. TME 10 Management Region Layout

In a one-way connection, the administrator at the managing server can control resources in the managed server TMR, but an administrator at the managed server has no access to resources in the managing server TMR. This can create a form of management hierarchy, but as management can only occur on directly connected TMRs, the hierarchy can only be one level deep, as illustrated in Figure 13.

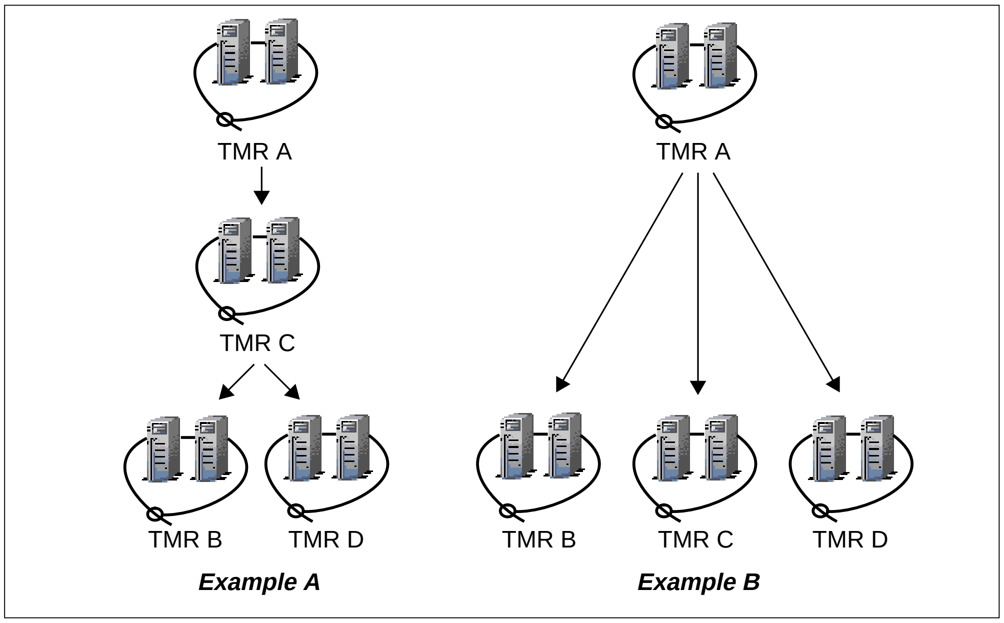


Figure 13. TMR Management Hierarchy

Example A in the above picture, shows an invalid TMR management structure. TMR A could be configured to manage TMR C, and TMR C can be configured to manage TMRs B and D, but in this setup, an administrator at TMR A has no control over TMRs B and D. ***Example B*** shows a valid single-level hierarchy, where an administrator at TMR A could control resources in TMRs B, C, and D. You might also picture a single-level hierarchy as a hub and spokes configuration, where one system acts as a hub (TMR A in ***Example B***), and the others are all spokes fanning out from the hub. The administrator only needs to connect to the hub to manage all the systems in the wheel.

In a two-way connection, administrators with the correct roles can manage resources in either TMR. See also Section 4.3.2.1, "TME 10 Administrator Roles" on page 47, for information on what happens to TME 10 administrator roles when TMRs are connected.

Note

As TMRs are joined, the data in the name registry of each TMR is optionally mirrored to allow cross-TMR administration. Even when connected, a change in the name registry of one TMR is not reflected in the other. The updates must be initiated at an interval that will depend on the amount of changes taking place in each TMR. The update can be scheduled in a job to occur regularly.

4.3.2 TME 10 Framework Administrators

In order to perform the management functions provided within TME 10 products, the TME 10 Framework defines *TME 10 administrators*.

TME 10 administrators manage systems resources in the TMR. The tasks that can be performed by TME 10 administrators can be defined through roles and by the make-up of an administrator's TME 10 desktop.

When the TME10 Framework is first installed, the initial TME 10 administrator is defined. This administrator has root authority on UNIX systems and administrator privileges on NT systems. This user can then delegate authority to other TME 10 administrators who can perform system management tasks without requiring access to a super user UNIX password or the Administrator NT password.

4.3.2.1 TME 10 Administrator Roles

Actions available to a TME 10 administrator depend on which authorization *roles* the administrator has been granted. This is similar to the concept of granting operator privileges to IBM Warp Server users or adding users to operator groups in Windows NT.

The roles that are available depend on the TME 10 products that are installed. The default roles provided in TME 10 Framework allow control to be divided in to such functions as backup, restore, and install_product. Roles with a broader management scope include super, senior, admin and user—each providing different capabilities in the administration of the TME 10 environment. It is important to note that the roles are not hierarchical. This means, for example, that it is possible for an administrator to have *senior* capabilities but not be able to perform a backup if your policy requires such a configuration. This avoids the need to maintain multiple superusers.

The roles can apply at the TMR or resource level. Whatever resource the roles are applied to, the administrator, by default, will have the same role for all objects contained within that resource. For example, if an administrator has a particular role at the TMR level, then by default that administrator will have the same role for all resources in the TMR. The role for each subordinate resource can still be set to some other levels as required.

There are three stages to implementing a new TME 10 administrator's roles.

1. Determine what actions the administrator will need to take and on which types of resources.
2. Identify the resource role or TMR role required to enable the actions.
3. Assign the necessary resource and/or TMR roles to the administrator.

As an administrator with the administrators resource role and a role in the TMR of senior, you can create other administrators. During the creation of an administrator, you define their TMR and resource roles, notice group memberships and the set login names. Once an administrator is defined, you can determine what will appear on their desktop, either by dragging and dropping icons representing policy regions, task libraries and so on into their desktop or by using a command line equivalent (`wln`).

Refer to the *TME 10 Framework User's Guide* or to the individual TME 10 application user's guides for more information regarding administration roles and how to define administrators.

Note

When TMRs are connected, TME 10 administrator roles are mapped between the two TMRs. For example, if an administrator has the admin role in TMR A and it is connected to TMR B, then the administrator also has the admin role in TMR B. An administrator in TMR B will not have access to TMR A unless the connection is made two-way (see Section 4.3.1.1, “TMR Connection” on page 45).

This DOES NOT apply to the super administrator role which is not mapped.

4.3.2.2 TME 10 Administrator Logins

Each TME 10 administrator will have a record in TME 10 that contains a *user login name* and one or more *set login names*:

user login name	The administrator’s TME 10 name, sometimes referred to as the TME 10 Principal. This identifies the TME 10 roles assigned to the administrator and the TME 10 desktop that is used for the administrator.
set login name	The TME 10 administrator logs in to a system using the system-specific login name and starts the TME 10 desktop. A table of set login names maintained for each TME 10 administrator is used to match the system-specific login names to the TME 10 administrator name.

The set login name can be further qualified with a managed node name to limit which managed nodes the administrator may login from.

For example, defining the user root@node.airline.com in the set login names list for the TME 10 root administrator would mean that root logged in at node.airline.com would use the TME 10 root administrator’s desktop. It may also be that richardh (for instance, unqualified) was specified in the set log in names list for the TME 10 root administrator. This would mean that richardh could login at any managed node in the TMR and would receive the TME 10 root administrator’s desktop. It is therefore recommended that you do not specify any unqualified names. An unqualified name leaves open the possibility that if someone could add a machine to the network and define a user of the same name, then they could log on and get the TME 10 administrator rights assigned to that name.

Now suppose there were a second TME 10 administrator named PrinterAdmin, defined solely to manage one resource. If the set login name table for this administrator had richardh@nuke.islandia.com defined, then when richardh logs in at that node he would receive the PrinterAdmin TME 10 desktop (a qualified name takes precedence over an unqualified one).

Note

Neither an unqualified user name nor multiple identical qualified login names can appear in more than one TME 10 administrator’s set login names table.

4.3.3 User IDs Used by the TME 10 Framework

Once a TME 10 administrator has the icon available to start an action, the administrator must next have the required role in the policy region of the destination system (the endpoint). Then, most TME 10 applications allow the definition of which system ID will be used to perform an action on endpoints, such as executing a task or distributing a file package. One of three options can be chosen:

- The system ID of the administrator initiating the action
- A named system ID
- A named group ID

It is important to note therefore that whichever ID is used, it must exist on the endpoint, and it must have the necessary system permissions to complete the operation (see also Section 4.3.5, “Tasks and Jobs” on page 51).

The TME 10 Framework in a UNIX environment makes use of the nobody ID to perform certain operations in a managed node. One example of this would be a command executed as a response action resulting from a TME 10 Distributed Systems Monitoring (Sentry) threshold event. If the nobody user does not exist in UNIX environments, TME 10 adds the user account tmersrvd. In NT environments, the TME 10 Framework adds an account tmersrvd, an unprivileged user account in the *Domain Users* group. None of these IDs should be removed once TME 10 is in use.

4.3.4 Policy and Policy Regions

Chapter 2, “Security Concepts and Overview” on page 5, discusses the importance of building and maintaining a security policy for your organization. With the concept of TME 10 policies and policy regions, we begin to see how a good security policy can be effectively implemented.

As the number of system and network resources to be managed increases, it becomes more important to have some way of dividing up the management function into some logically abstracted pattern. TME 10 does this through management policies. A policy specifies management rules for resources of a particular type. For example, a policy can be implemented to determine that selected tasks can only be run on a specified host, where tasks and the hosts (nodes) are forms of managed resources. The resources that can be managed depend on the TME 10 applications installed. A TME 10 managed resource represents a system or network resource that you manage with TME 10. As an example, those provided with the TME 10 Framework are:

- Managed nodes
- PC managed nodes
- Task libraries
- Profile managers

The TME 10 Security Management product adds the *SecurityProfile* managed resource, which contains a number of subresource types such as groups, roles and resources.

The TME 10 products installed determine the resource types available although it is possible to create completely new managed resources with the TME 10 Application Developer’s Environment (ADE) toolkit.

Policies are implemented in a scalable fashion through the use of *policy regions*. Resources to be managed such as nodes and TME 10 resource types are assigned to a policy region. Administration activities on that resource will then be subject to a *default policy* and possibly a *validation policy* for that resource. A default policy defines a set of default values (*properties*) that are assigned to a resource when that resource is created. For example, when you create an item from the *UserProfile* resource provided in the TME 10 User Administration product, the default policy presets certain user properties such as password rules. The defined default policy can be one of three types:

None	No default value.
Constant	Default is set as a constant.
Script	Default is set by a script.

Validation policy determines whether all resources in a policy region *must* comply with the region's established policy. Enabling a validation policy prevents TME 10 administrators from creating or modifying resources in a way that does not conform to the validation policy of the region in which the resources are created or located. For a profile-based resource, a validation policy can be qualified to apply to individual properties of a particular profile item. The defined validation policy can be one of three types:

None	No validation value.
Constant	Validation is set as a constant.
Script	Validation is set by a script.

Please see Section 5.11, "Default and Validation Policies" on page 91, for an example of how this is implemented in TME 10 Security Management.

A further policy region refinement allows for the hierarchical subdivision of policy regions into *policy subregions*. Each policy region can contain either resources being managed or other policy regions, dividing up the region into smaller chunks. The top-level policy region for a particular resource type might specify one set of properties and subregions initially inherit the policies defined in the parent region. After that, a subregion policy might alter the properties to match a specific requirement.

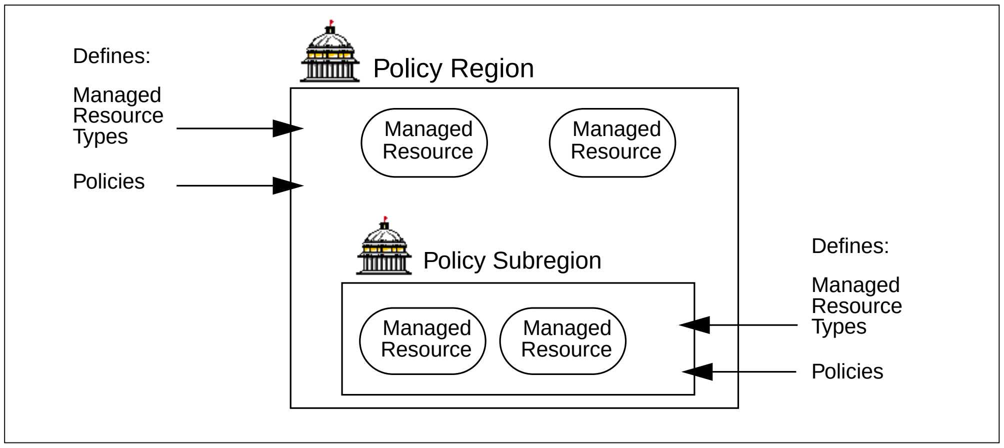


Figure 14. Policy Region with Policy Subregion

See the *TME 10 Framework User's Guide*, Policy and Policy Regions section for more details on this subject.

4.3.5 Tasks and Jobs

TME 10 tasks and jobs allow you to set up a procedure for an operation once, then run the procedure on selected machines whenever necessary. TME 10 defines tasks and jobs as follows:

Task	A definition of something that needs to be done to the network on a routine basis, such as clearing the printer queue or some other TME 10-supported operation.
Job	A job is a task that is executed on specific managed resources.

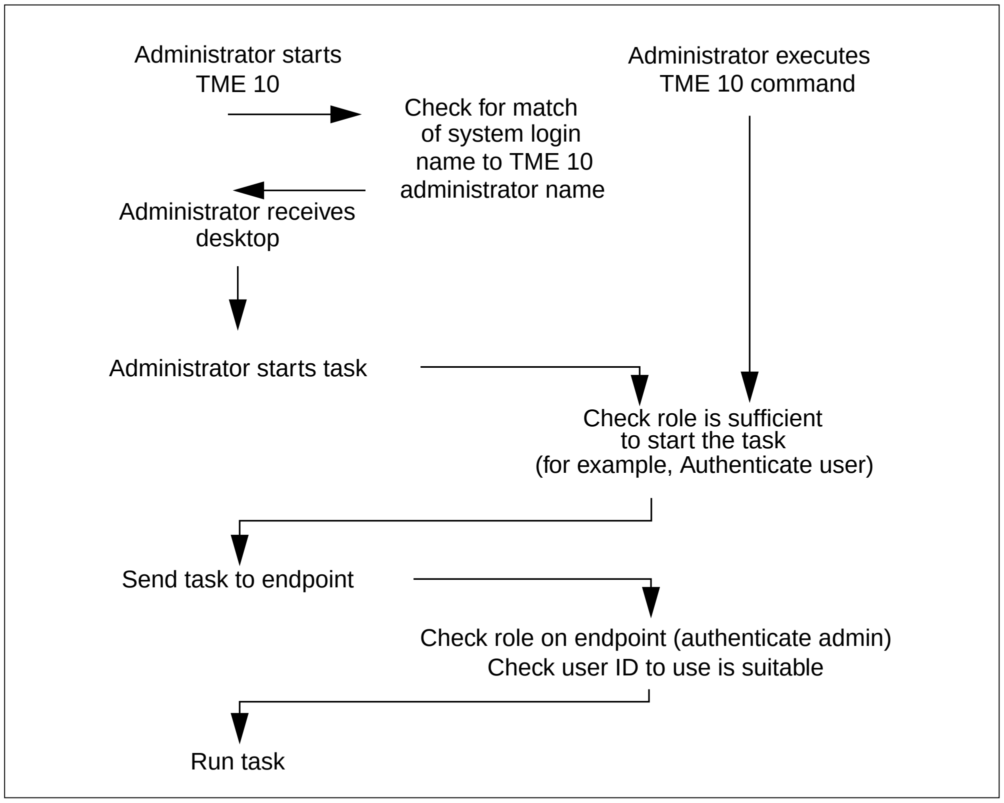
Among the details you specify when defining a task are the role required to execute the task and the user or group ID under which the task will be run. The administrator starting the task must have the correct role in the policy region of the task endpoint. The default task properties specify the user ID used to run the task as the ID of the administrator that runs the task. The default setting for the group ID is nogroup.

Note

By default, TME 10 prevents you from defining root as the user or group ID *required* to run a task. However, it is possible to modify TME 10 validation policies to allow this if required. See the *TME 10 Framework Reference Manual* for procedures on editing policy for tl_val_set_uid or tl_val_set_gid. It is also possible for root to run a task if no ID was specified in the task properties.

The set login names for an administrator allow the administrator to use any one of a number of system login IDs from any supported system type to perform TME 10 administrator functions. This provides the matching between different user IDs from different systems (such as daffy on one system and daffyd on another) to a single administrator for TME 10 functions. Now suppose you want to execute a task or some other TME 10 function on a managed node where your ID is different from your user login name defined to TME 10. TME 10 provides another way of mapping multiple user names to a TME 10 administrator called ID mapping. ID mapping is done on system type (for example NT or UNIX). So you may define a UNIX entry of bobs and an NT entry of Bob_Smith. Now, when the task is run on a UNIX system, it will run under the ID bobs, and on an NT system it will run under the ID of Bob_Smith. See the `widmap` command in the TME 10 Framework documentation for details of how to ad ID map names. When you have ID mapping set up, you tell TME 10 to check the ID map by specifying the administrator’s TME 10 user login name preceded by the dollar sign ($).

The following figure shows the flow employed when a TME 10 administrator runs a task.



Administrator starts TME 10
Administrator executes TME 10 command
Check for match of system login name to TME 10 administrator name
Administrator receives desktop
Administrator starts task
Check role is sufficient to start the task (for example, Authenticate user)
Send task to endpoint
Check role on endpoint (authenticate admin) Check user ID to use is suitable
Run task

Figure 15. TME 10 Administrator Task Implementation Flow

4.3.6 Notices

An important part of any security policy is the tracking of administrator actions. TME 10 incorporates a notification facility that informs administrators of system management operations detailing which administrator performed the operation.

TME 10 implements this audit trail function through the use of *notices* and *notice groups*:

notice	A message concerning some operation or change in the distributed system.
notice group	A collection of notices specific to a TME 10 application or operation. TME 10 administrators subscribe to notice groups to receive messages on their TME 10 desktop notice boards.

Messages are stored in a language-neutral format. Different operators can see the same notice in different languages if required.

4.4 TME 10 User Administration

TME 10 User Administration provides a range of user management functions. This section includes a short discussion of the security features utilized by this product. This also serves as an example of how TME 10 products operate in a secure manner, often making use of the security-related features of the TME 10 Framework.

See Section 8.5, “Integration with TME 10 User Administration” on page 188, for some information on plans for tighter integration between TME 10 Security Management and TME 10 User Administration.

The main topics discussed in the remains of this chapter are the ways user passwords are handled by TME 10 User Administration. Refer to publications such as the TME 10 User Administration *User and Group Management Guid*e or the redbook titled *Getting Started with TME 10 User Administration*, SG24 2015, for more information about User Administration.

4.4.1 Password Storage in User Profiles

When you define or alter a user in TME 10 User Administration, the password for the user is encrypted before being stored in the TME database. The method of encryption used varies by target system:

UNIX:	The password is encrypted using the UNIX one-way encryption.
Other Platforms:	The password is encrypted using the TME 10 Framework encryption process, sometimes referred to as the AES tas_encrypt() function.

4.4.2 Password Population and Distribution

TME 10 User Administration can populate the User Administration profiles database with records for users that are already defined in managed systems: RACF, UNIX, NetWare (NDS or Bindery), and Windows NT. Since UNIX passwords are accessible as a string representing the encrypted password in the /etc/passwd file, UNIX users being populated into a user profile can have their passwords read into the profile, too. Subsequent distributions of those users to UNIX systems then simply add the encrypted password to the /etc/passwd file when adding the user. Windows NT and NetWare passwords cannot be read into user profiles during population, and by default, the password is set to match the login name during population of a new user. If a user exists on more than one system from which population data is being read, whether information is overwritten depends on a choice made by the administrator prior to population.

When user profiles are distributed, user passwords are only overwritten on endpoints if the password itself was modified in the profile (or if `wpasswd -l` is run). If other settings in the profile are changed, the password at the endpoint will not be reset by a distribution. For UNIX end nodes, the UNIX encrypted password is sent during distribution. For other end nodes, the tas_encrypt() password is sent, and when the password reaches the end node, it is decrypted by TME 10 and re-stored using the end node standard encryption method.

See Section 4.2, “Network-Level Security of the TME 10 Framework” on page 33, for details on TME 10 Framework encryption and data transmission.

4.4.3 TME 10 User Administration Password Commands

There are several ways that a user’s password can be changed:

- TME 10 User Administration `wpasswd` command
- TME 10 User Administration `wsetusr` command
- TME 10 User Administration `wsetusrs` command
- Native operating system command or tool

See the TME 10 User Administration *User and Group Management Guid*e for the syntax of the `wpasswd`, `wsetusr` and `wsetusrs` commands. The user or an administrator may use `wpasswd` to alter a user's password. Since `wpasswd` is more likely to be used, we discuss it a little further here.

The `wpasswd` command allows a user or TME 10 administrator to change the user's password. An attempt is made to change the password on all TME 10 User Administration profiles that contain a record for the user. There is an argument (`-l`) that also changes the password locally for the user.

There are arguments that allow you to change the UNIX, Windows NT, NetWare, and TME 10 common passwords, individually. If you do not specify any of these options, the `wpasswd` changes UNIX passwords only.

Note

The `wpasswd` command is not intended to be a replacement for the UNIX `passwd` command. The `wpasswd` command will not run if the oserv process is not running on the local host or if the TMR server is down. You may need to change UNIX passwords when a host is in "single-user mode" or at other times when the oserv process is down.

The `wpasswd` command does not initiate a distribution of the user profile records that have been changed—it is not really a way of coordinating passwords across systems. This allows users to update their passwords quickly and lessens contention among users for password updates to the same user profile. We currently rely on other products to synchronize passwords across systems or by direct action of the user or administrator.

The `wpasswd` command can fail if a user is defined differently in two or more different user profiles. For example, if a user is allowed to change the password in one profile but cannot change the password in a second profile, the password change will not register in the second profile. When both profiles are distributed, the user's password on the hosts that subscribe to the second profile will be different than those that subscribe to the first profile.

The user password is treated as a form of special property in the user record. If you allow the user to control his or her password and you or the user change the user's password (for instance, with the `wpasswd` command), then when you distribute the user profile containing this user's record, the user's password is updated on all the specified endpoints. The rest of the record is not distributed. If some other property of the user record is changed and the password remains the same, the password is not redistributed to the end points. This behavior is different from other properties in the user record. If any other property in a user record is changed, the whole record is distributed along with changed and unchanged properties.

4.4.4 Operating System Password Support

If the password stored in a user profile is to remain the same as the password at individual systems, then users will need to run the `wpasswd` command when they change their password. This command is available only on managed nodes and not on PC-managed nodes.

Otherwise, the user uses whatever mechanism is normally available for changing the password on a particular system. However, if you want synchronized passwords across all systems, then you would disallow users to change their password on PC managed nodes. As long as no local changes of passwords occur, profile distributions of TME 10 user profiles are able to update the passwords on the target systems.

The reach of TME 10 User Administration (and TME 10 Security Management) can be extended through products such as the Santix DCEmgmt suite of products, bringing the same capabilities to DCE environments, and the TME 10 Module for Domino/Notes, which extends the support to Lotus Notes and Domino servers.

Attention

There are special considerations for maintaining user passwords when TME 10 Security Management password controls are in place—in particular on UNIX systems. In the first product releases, the TME 10 User Administration `wpasswd` command is not subject to the TME 10 Security Management password validity checks. Some invocation of the TACF `sepass` utility must take place for that to happen.

Chapter 5. Understanding TME 10 Security Management

TME 10 Security Management provides the following features for security management across multiple platforms:

- Access control to resources based on job functions
- Systemwide security polices for passwords, audit and logins
- Common interface to different security systems
- Audit capabilities
- Integration with existing TME 10 applications and Framework services

In this chapter, we discuss the concepts and architecture of the TME 10 Security Management product in the following sections:

- Architecture and endpoint support
- Role-based resource access administration
- Security profile records for groups, resources and roles
- Security profile records for system policy
- Default and validation policies
- Integration into TME 10 Framework
- Integration with other TME 10 products
- TACF

5.1 Overview and Product Information

The purpose of this section is to give you a high-level view of how the product is structured, introduce its terms and features, and list the supported platforms.

5.1.1 TME 10 Security Management Architecture

TME 10 Security Management is a profile-based application. As such, it utilizes standard TME 10 Framework features such as:

- Profile managers
- Endpoint subscription
- Distribution
- Notice groups
- Task libraries

Figure 16 illustrates the architecture and how the above-listed features work together.

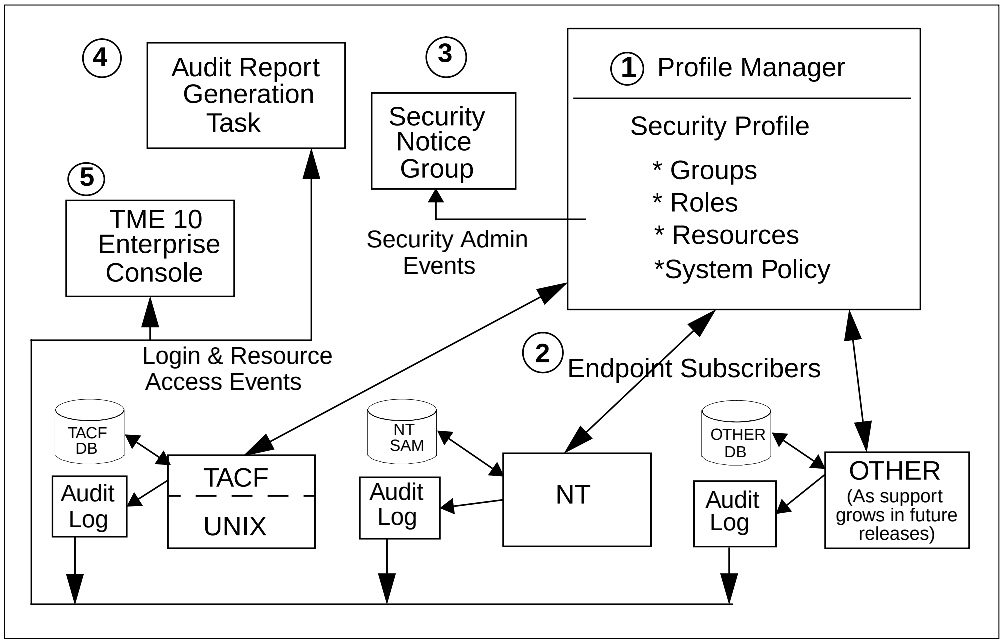


Figure 16. TME Security Management Architecture

The numbers in the list below refer to the numbers on the figure:

1. Security profiles can be created in profile managers in the same way as user and group profiles for TME 10 User Administration.

2. NT and UNIX systems can subscribe to the profile managers containing the security profiles.

   - The two-way link between the profile manager and endpoints indicates that data can flow in both directions. Security records can be created in the security profiles and distributed to the endpoints, and data can be collected from the endpoints to populate security records.

   - Each of these endpoint types has its own native security system, the NT Security Account Manager database and TACF on UNIX, into which security information from the security profiles is mapped.

3. Notice groups contain information about activities performed by administrators related to specific functions. For example, there are notice groups for the scheduler and for TME 10 User Administration. Notice groups can then be made available to the TME administrators who have authorization to perform functions in those areas. The security notice group contains information on security events such as the creation or modification of security records and the distribution of security profiles.

4. The audit report generation task, plus many other tasks supplied with TME 10 Security Management, uses the TME 10 Framework task library services. You can run tasks and jobs (which use the TME 10 Framework scheduler) on specific endpoint systems or via profile managers to the profile managers' subscribers.

   Audit information is generated at the endpoints depending on the audit control attributes that you specify and can be collected from those systems by the audit tasks.

5. Audit logs on the endpoint systems can be monitored for specific events. Details of these events can then be routed to the TME 10 Enterprise Console where actions can be triggered or operators alerted to possible security problems.

5.1.2 Platform Support

TME 10 Security Management needs the TME 10 Framework at Version 3.1 or above. Earlier versions of TME 10 must be upgraded before TME 10 Security Management can be installed.

In the first release, TME 10 Security Management is supported in the following environments (check the release notes shipped with the product to ensure the platform you will be installing on is supported):

- TME 10 Desktop Console support for:
  - AIX 3.2.5, AIX 4.x
  - HP-UX 9.04, 9.05, 10.01, 10.10
  - SunOS 4.1.3, 4.1.4
  - Sunsoft Solaris 2.3, 2.4 2.5
  - NT 3.5, 3.5.1, 4.0
  - Windows 3.1 or higher
  - Windows 95
- TME 10 Server support for:
  - AIX 3.2.5, AIX 4.x
  - HP-UX 9.04, 9.05,10.01,10.10
  - SunOS 4.1.3, 4.1.4
  - Sun Solaris 2.3, 2.4, 2.5
  - NT 3.5, 3.5.1, 4.0
- TME 10 Managed Node support for:
  - AIX 3.2.5, AIX 4.x
  - HP-UX 9.04, 9.05, 10.01, 10.10
  - SunOS 4.1.3, 4.1.4
  - Sun Solaris 2.3, 2.4, 2.5
  - NT 3.5, 3.5.1, 4.0
- DSL-based GUI for Windows and Motif

5.2 Endpoint Support

TME 10 Security Management 3.2 (the first full release) provides support for distributing its security profiles to the following endpoints:

- UNIX (see above for supported UNIX versions)
- Windows NT domains

5.2.1 UNIX Endpoint

It is generally agreed that UNIX has rather limited security features; so TME 10 Security Management provides a low-overhead component on the UNIX system that implements checks for accesses to system-level resources. These are a seamless addition to those normally provided by UNIX security. This component is the Tivoli Access Control Facility (TACF).

The TACF implementation is the same for all versions of UNIX that are supported, providing a consistent level of function. TACF intercepts attempted access to resources and grants or denies access depending on information held in its database. TME 10 Security Management updates the TACF database through the distribution of security profiles.

A TACF Command Line Interface is provided for local system override. Refer to Section 5.16.3, “TACF Commands” on page 112, for more details of this interface. It is not likely that you will need to employ the command line interface for general management tasks, only if you wish to perform out-of-the-ordinary functions. To do normal security management functions from the command line, you use the TME 10 Security Management commands that are added to the TME environment such as `wcrtsec` for creating a record in security profile. See Chapter 7, “Using TME 10 Security Management” on page 145, for examples of the use of some of these commands. The syntax of the commands is given in the *TME 10 Security Management User's Guide*.

TACF is discussed in more detail in Section 5.16, “TACF” on page 105.

5.2.2 NT Endpoint

An interface component is provided to interface directly with the NT security related APIs. When TME 10 Security Management distributes security profiles to an NT system, the TME 10 Security Management NT component directly interfaces with the NT Security APIs to update the NT security structures. Unlike TACF on UNIX, no additional security code is required—simply the mapping of the distributed profile to the NT security function.

Note

TME 10 Security Management creates a number of extra items in NT systems, similar to those created in TACF, in order to perform the management tasks. For example, there is an entry in the NT Registry under `HKEY_LOCAL_MACHINE\SOFTWARE\Tivoli\TME\SECMGT Roles`. Also, the product manages an NT domain user account for each NT global group that is managed through a security group. This is given the name `tme_<security group name>`. None of these management resources should be tampered with using local NT tools.

5.3 Security Profiles

We have mentioned that TME 10 Security Management is a profile-based application. This provides function for the following:

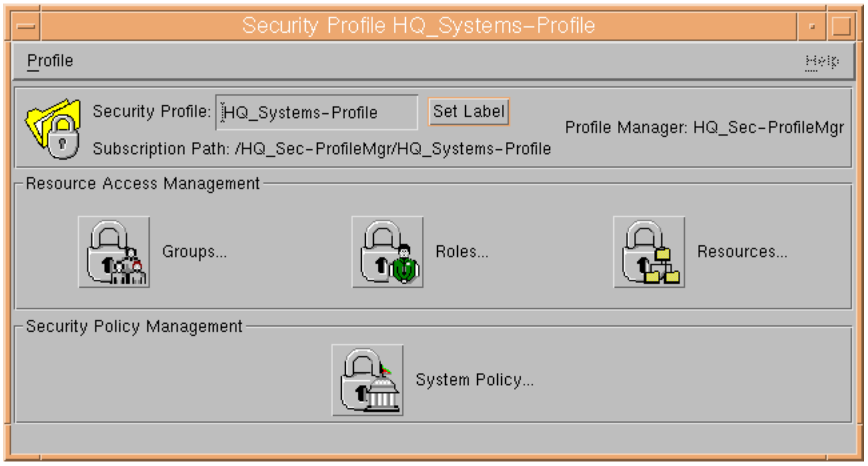
- Assigning security profiles as managed resources in a policy region.
- Creation of security profiles in profile managers. The Security profiles contain records that define security related information.
- Subscription of profile managers and/or managed nodes to a profile manager containing a security profile.
- Default and validation polices.
- Populating security profiles.

- Distribution of copies of the security profile to its subscribers, where the system-independent format of the records is mapped onto the system-level attributes

Security profiles are slightly different from other profiles that you might be familiar with, such as user and group profiles in TME 10 User Administration. A user profile has a single type of record, where each record defines a single user. Security profiles differ in that they contain four different types of records:

- Group record
- Resource record
- Role record
- System policy record

When you open a security profile from the TME desktop, you will see this:


Security Profile HQ_Systems-Profile

Security Profile: HQ_Systems-Profile

Subscription Path: /HQ_Sec-ProfileMgr/HQ_Systems-Profile

Profile Manager: HQ_Sec-ProfileMgr

Figure 17. Four Types of Record in a Security Profile

Group, resource and role records provide the components for role-based resource access management. System policy records are provided for security policy management.

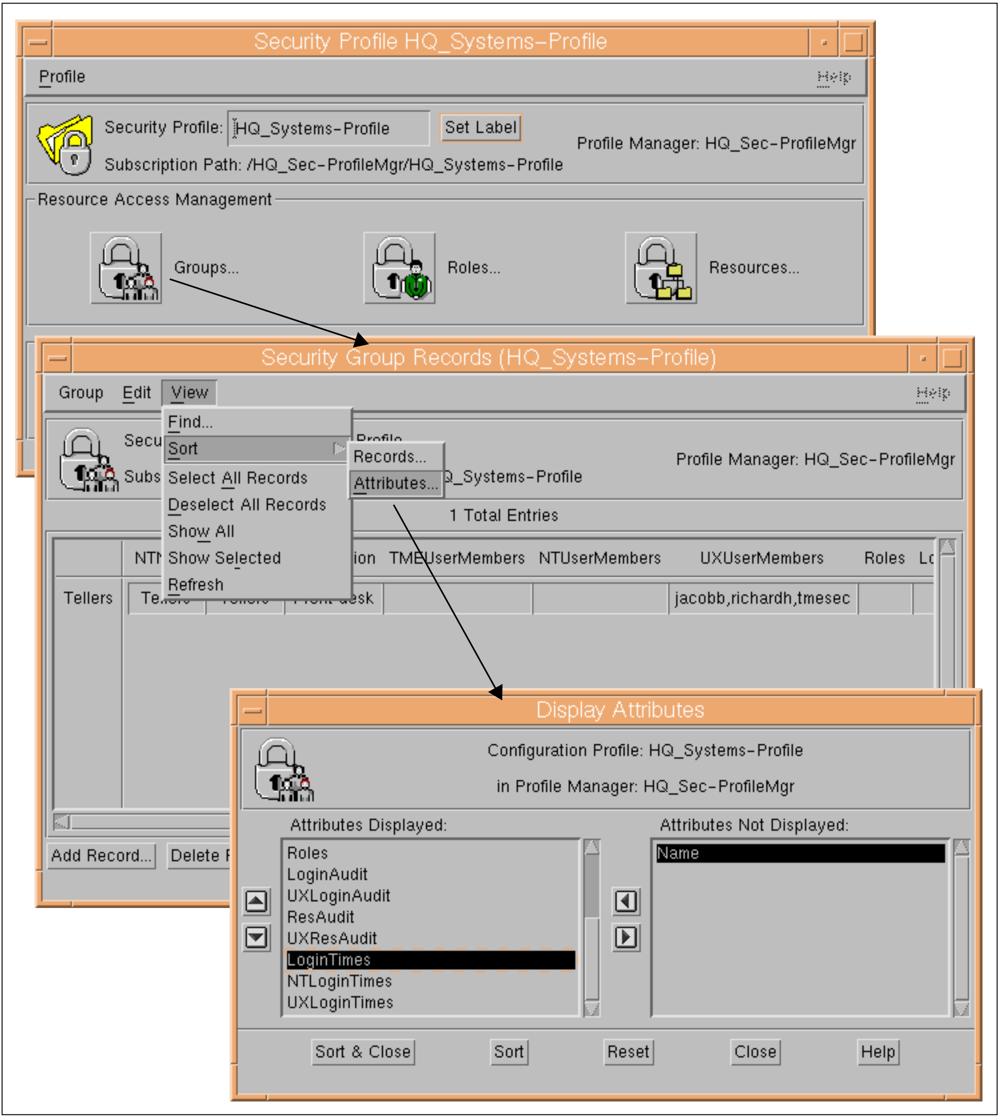
The names group, resource and role can be used in several different contexts. In the following sections, we use the following terms when referring specifically to TME 10 Security Management record types:

- Security group
- Security role
- Security resource

Security records have attributes (or properties) associated with them. Each record type has its own set of attributes. For example, the security group record has attributes for the TME, UNIX and NT user members of the group.

- TMEUserMembers
- UXUserMembers
- NTUserMembers

This is how you can display the attribute names for security group records:



Security Profile: HQ_Systems-Profile

Subscription Path: /HQ_Sec-ProfileMgr/HQ_Systems-Profile

Profile Manager: HQ_Sec-ProfileMgr

1 Total Entries

NTI	ion	TMEUserMembers	NTUserMembers	UXUserMembers	Roles	Lc
Tellers	Tellers	Front desk			jacobb,richardh,tmesec	

Configuration Profile: HQ_Systems-Profile

in Profile Manager: HQ_Sec-ProfileMgr

Attributes Displayed:

Roles
LoginAudit
UXLoginAudit
ResAudit
UXResAudit
LoginTimes
NTLoginTimes
UXLoginTimes

Attributes Not Displayed:

Name

Figure 18. Displaying the Attributes for Security Group Records

You see that there are many attributes that appear very similar. For example:

- LoginTimes
- UXLoginTimes
- NTLoginTimes

When you create or modify records, you are often given the option of setting the values of attributes as *global* or *endpoint-specific (UNIX / NT).* In this case, the global value is contained in *LoginTimes* and the endpoint-specific values contained in attributes of the same name, but prefixed with *UX* for UNIX or *NT* for NT. Here is a more detailed look at global and endpoint-specific attributes:

1. *Global Attributes*

Setting global attributes allows you to set the same policies and standards across all endpoint types with a single distribution of the records. Remember, however, that the attributes you see displayed when selecting a global option may not apply on all the endpoint types. For example, in the security group record, audit control for login attempts (LoginAudit attribute) is not supported on NT.

When the records are distributed, only the values that are supported will be applied to each endpoint type. See “Distributing the Security Profiles” on page 95 for details of which attributes are supported on each endpoint type.

Setting global values for attributes automatically propagates those values into the endpoint-specific attributes. If you do not supply values for the attributes, they are generated automatically from the default policies associated with the record. Default policies are discussed later in “Default and Validation Policies” on page 91.

2. ***Endpoint-Specific (UNIX, NT) Attributes***

   You may want to define different values for attributes on each endpoint type. For example, you may want to set a systemwide password policy in the TME system policy that applies to all endpoints, but then tailor UNIX endpoints to specifically change some of the attribute values. You could specify globally that inactive user accounts should be suspended, but decide that this should not apply on UNIX. You then modify the UNIX-specific values to reflect this. Your global values will still apply to NT.

   Where attributes are not applicable on a specific endpoint type, the field either does not appear on the dialog or is grayed-out. See “Distributing the Security Profiles” on page 95 for details of mapping attributes to the endpoint types.

We do not repeat here all the concepts of profile managers, profiles, subscription, and distribution. These are described fully in the *TME 10 Framework Planning and Installation Guide*, as well as in other redbooks, such as in *Understanding TME 10*, SG24-4948.

However, you will find the following functions described in more detail to show how they are implemented specifically in TME 10 Security Management:

- “Default and Validation Policies” on page 91
- “Populating the Security Profiles” on page 92
- “Distributing the Security Profiles” on page 95

In the following sections, we cover these topics:

- Role-based resource application in “Role-Based Resource Access Administration” on page 63
- Security groups in “TME Security Groups” on page 65
- Security resources in “TME Security Resources” on page 68
- Security roles in “TME Security Roles” on page 73
- Security System policy records in “TME System Policy” on page 83

5.4 Role-Based Resource Access Administration

Figure 19 shows a high-level view of TME 10 Security Management role-based resource access.

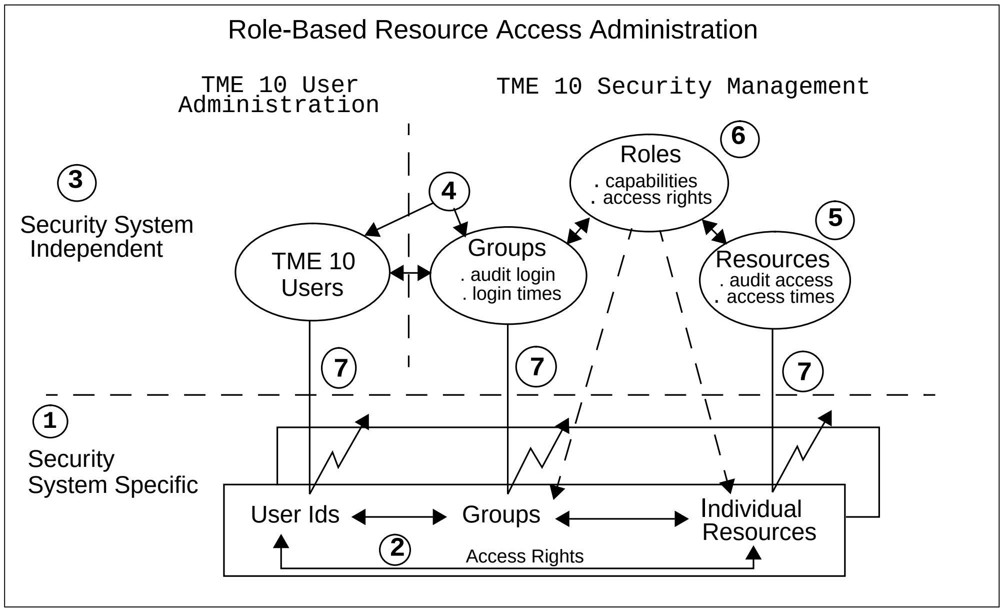


Figure 19. Role-Based Resource Access.

The numbers below refer to the numbers in the figure above:

1. At the system level, there are specific security systems native to each endpoint type. The endpoint types supported by TME 10 Security Management are:
   - UNIX Security
   - NT SAM/Registry
2. Each endpoint type has its own versions of users, group and resources with access rights defined for users and groups to the resources. Resource types, naming conventions, syntax of commands, relationships between users, groups and resources may all be very different.
3. TME 10 Security Management provides a security system that is independent of the system-specific security. It allows information that is relevant to all the endpoint types to be held in an independent format that can be later translated into system-specific security data at each endpoint type.
4. TME security groups can be populated with group information that already exists on UNIX and NT endpoints. Groups can also be created that do not currently exist at the system level. Users are assigned as members of one or more security groups. The users may be existing members of the system-level groups, or can be users defined in user profiles created by TME 10 User Administration, if that product is installed.

   Login time restrictions and audit attributes can be set for users in the security group.
5. Resources such as files, terminals and TCP services can be defined with default access permissions, audit attributes and access time restrictions.
6. Roles provide the link between the groups (and the users in the groups) and the resources. Roles are assigned resources and specific access permissions to those resources. Groups are assigned roles, thus giving the users in those groups access to the resources assigned to the roles.

7. Groups, resources and roles are distributed to the endpoint systems where they are mapped onto the native security system of that endpoint type.

In the following sections, we describe in detail the security groups, security resources and security roles. Each section contains references to the others; so you will need to read all of them (probably several times!) to fully understand the relationships between them.

We do not show in detail how to create and modify security groups, security resources and security roles. That sort of facility will be demonstrated in Chapter 7, “Using TME 10 Security Management” on page 145.

5.5 TME Security Groups

Groups are made up of people who have specific jobs to perform and functions to execute and therefore require access to the same resources. So you may set up security groups that mirror the business organization, for example:

- Divisions
- Departments

or that define cross-organizational functions, for example:

- Generic job titles, for instance, managers
- Project teams

Groups defined in your native security systems on the various endpoint types should already reflect your organization; so you may choose to create your security groups by populating them from the existing system-level groups. See Section 5.12, “Populating the Security Profiles” on page 92, for more details.

Once the security groups are set up and given access permissions to resources through security roles, they are unlikely to need many changes other than adding and removing members of the group as people join, leave and move within the company.

The information held in security group records includes:

Group name	Can be a global name or endpoint specific
Member list	Users who belong to the group
Audit control attributes	Define how login and resource accesses will be logged
Login time restrictions	Define the time of day and days of week that users in the group are allowed to login
Roles	Roles that are assigned to the group

The attributes (properties) associated with the information held in the records may not be applicable on every endpoint. See Table 7 on page 98 for details on how these attributes map to the specific endpoints.

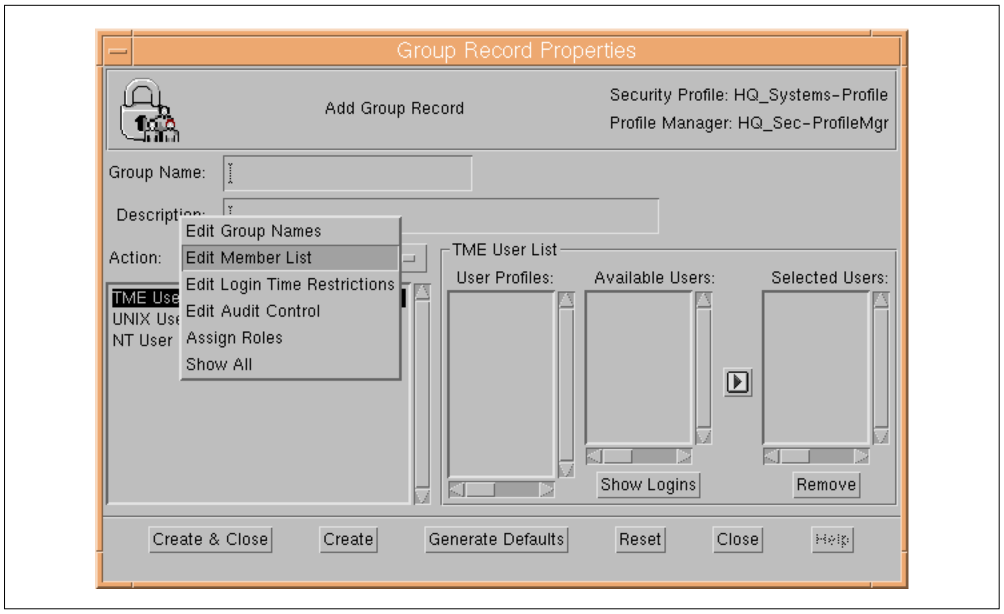


Figure 20. Dialog for Adding a Group Record

This figure shows the dialog panel that is displayed for adding a security group record. The information that is held in security group records is described in the following sections.

5.5.1 Group Name

The syntax and naming conventions of groups varies between endpoints. For example, the maximum length or the use of upper and lower case.

With TME 10 Security Management, you can specify a global name that is valid for all the target platforms, or you can specify a global name plus endpoint-specific names. Any endpoint-specific value will override the global value for that particular endpoint type.

5.5.2 Member List

You can select members to add to security groups from lists of UNIX and NT users from all your managed nodes and from TME user profiles if TME 10 User Administration is installed. The dialog panels present you with a list of managed nodes for UNIX and NT or a list of TME user profiles. Selecting a UNIX, or NT managed node shows you a dynamically generated list of users on that managed node. Selecting a TME user profile shows you a list of users in that profile.

When you add a member from a TME user profile, the security group record maintains a list of endpoint-specific login names for the user you select. For example:

> You may have a user record in a user profile with a common login ID of *bclinton.*
>
> This login will, by default, be used as the login ID for the UNIX and NT systems to which this record is distributed.
>
> You may, however, have changed the specific UNIX login to a different name, for example *bill*.

When you display the user profile to select users to add to a security group, *bclinton* is the user name that you will see and select, but the UNIX login name *bill* is also stored in the record. This is required so that when the records are distributed, it is the UNIX login ID of *bill* that is added to access control lists of resources rather than the name *bclinton*.

Users can be members of multiple security groups. See “Example 1: User in Multiple Security groups” on page 76.

There is no default for this field.

5.5.3 Login Time Restrictions

Day-of-week and time-of-day login restrictions are supported. These are very similar to the access time restrictions for security resources. You can specify all days, weekdays or specific days and start and stop times.

Time restrictions can be set at a global level and at the level of specific endpoints. Endpoint-specific restrictions override global restrictions.

The default is *Anyday @ Anytime*.

5.5.4 Audit Control

You can specify that audit records are logged for the following:

1. Login attempt failures and successes
2. Successful or failed access to resources

Audit attributes can be set as follows for each security group:

FAILURE	Failed accesses to resources
SUCCESS	Successful accesses to resources
NONE	nothing
LOGINFAILURE	Login failure of users in security group
LOGINSUCCESS	Login success of users in security group

If a user is in more than one security group, the effects of the audit attributes are cumulative. For example, if a user is a member of security group Managers, where LOGINFAILURE is set, and is a member of security group Support_Users, where LOGINSUCCESS is set, then audit events are logged for every login failure and success. If the audit attribute is set to NONE in one group and LOGINFAILURE in another, failed logins are logged.

You will see in “TME Security Resources” on page 68, that audit control attributes for accessing resources can also be set at the resource level.

If an attempt to access a security resource is made by a user who is a member of a security group, the cumulative audit attributes of the security group and security resource apply. This applies whether or not the security resource is explicitly available to the user through its assigned roles. In effect, both the user and the resource are being audited.

Audit attributes can be set at a global level and at specific endpoint levels. Endpoint-specific audit attributes override global audit attributes.

The defaults are *Failure and Loginfailure*.

5.5.5 Roles

You will see a list of security profiles that are available and by selecting one, you can add the security roles defined in that profile. You can select multiple security roles to assign to the group.

The users defined in the member list of that security group then have access capabilities to all the security resources assigned to those security roles. You will see later that security roles can also be assigned to security groups from the security role interface.

5.6 TME Security Resources

Resources are objects in your distributed environment that need to be protected, but you do not need to protect everything. Be selective. Define resources that in general nobody should be able to access. Then you can specify a default access of *none* and make those resources available only to the users who really need them through the security roles and their access permissions.

Be careful about creating resources that have to be available to everybody. You might set default access levels too low and prevent all users from accessing them.

Security resources can be created for objects such as:

- Files
- Processes
- Printers
- Terminals
- TCP/IP client services

The system resources that are defined as security resources are endpoint specific. There is such variety of resource types across the endpoint types that it is not feasible to define common resource types that can be mapped onto the endpoint types. Each security resource maps to a specific endpoint type, a resource type valid for that endpoint type and the system resource name. For example, there might be a security resource for UX:FILE:/etc/passwd and another for NT:DIRECTORY:C:\WINNT.

Security resources can be logically grouped together by associating them with security roles that relate to specific job functions within an organization. Security resources can be associated with more than one security role, with each security role having a different level of access to the security resources. See Figure 27 on page 80.

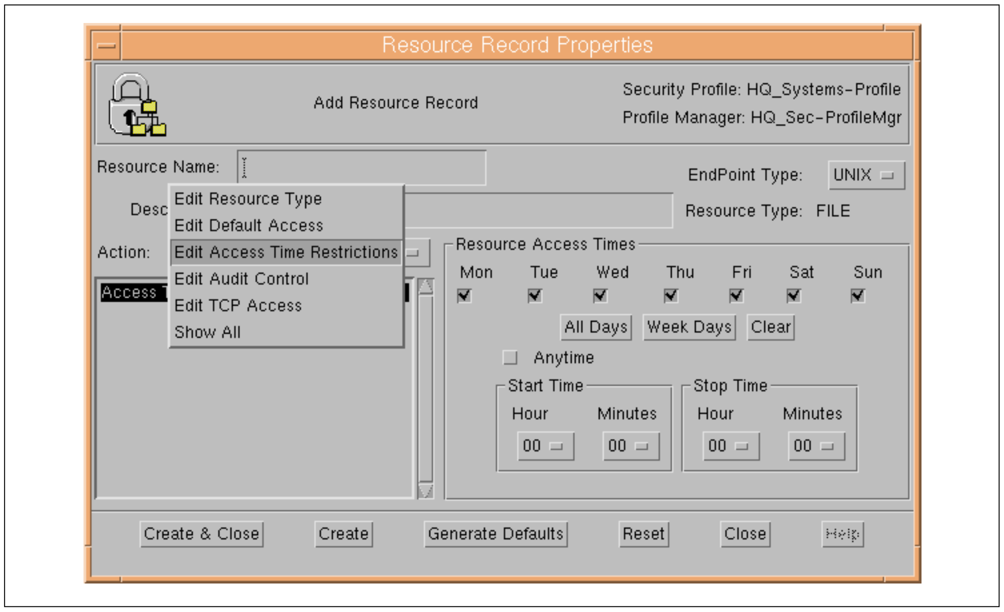
Security resource records contain access information for files, directories, printers, and so forth that exist or are defined on your systems. The information is stored in the record in a format independent of system type, and it is mapped onto the specific endpoint resource type and resource name when the records are distributed.

The information held in security resource records includes:

Resource specification	Consists of the endpoint type, the resource type and the resource name

Access time restrictions	Specifies periods in which resources can be accessed
Audit control attributes	Determines which access attempts should be logged
Default access	Defines the access that users are granted where they do not have explicit access permissions defined through a security role
TCP service access	Defines which hosts or IP addresses have access to TCP services
Role list	Contains the names of the security roles that have the security resource record assigned to them

The attributes (properties) associated with the information held in the records may not be applicable on every endpoint. See Table 8 on page 99 for details on how these attributes map to the specific endpoints.


Resource Record Properties

Add Resource Record

Security Profile: HQ_Systems-Profile
Profile Manager: HQ_Sec-ProfileMgr

Resource Name:
EndPoint Type: UNIX
Resource Type: FILE

Resource Access Times: Mon ✓ Tue ✓ Wed ✓ Thu ✓ Fri ✓ Sat ✓ Sun ✓

Anytime ☐

Start Time: Hour 00, Minutes 00
Stop Time: Hour 00, Minutes 00

Figure 21. Dialog to Add a Security Resource Record

This figure shows the dialog panel that is displayed for adding a security resource record. The information that is held in security resource records is described in the following sections.

5.6.1 Resource Specification

The resource specification consists of an endpoint type, a resource type and a resource name. The endpoint type is either UNIX or NT. Resource types and names are covered in the following sections.

5.6.1.1 Resource Type

The number and type of resources vary across the systems that are being managed. Here is a list of the resource types that are supported on the current endpoint types:

UNIX

- CONNECT - potential network servers that users and/or groups of the subscribed systems will connect to.
- FILE - defines discrete and generic files and/or directories with their access rules.
- PROCESS - programs or applications, running in their own address space that should be protected from being killed.
- PROGRAM - programs that are considered part of the trusted computing base.
- SECFILE - similar to PROGRAM but SECFILE records cannot appear in a conditional access list.
- SURROGATE - restrictions to protect users from the su command.
- TERMINAL - potential terminals of subscribed systems. Users can sign on from a terminal only if they have been authorized to use the terminal.
- TCP - services defined in /etc/services can be protected from network hosts requests.

NT

- FILE - NTFS files
- DIRECTORY - NTFS Directories, files and directories below them
- SHARE - network accessible files and directories
- PRINTER - printers to be protected
- REGISTRY - the NT registry
- SYSTEM (NT USER RIGHTS) - defines the NT rights granted to a user
    - NT User Rights are slightly different from the other NT resource types. The entries can be created by populating security resource records from an NT system, but they cannot be edited. NT User Rights can be assigned to security roles. User rights supported are:

Shutdown (S)	Shut down the system
Remote Shutdown (RS)	Force shutdown from a remote system
Remote Access (RA)	Access this computer from the network
Manage Domain (MD)	Add workstations to domain
Manage Audit (MA)	Manage auditing and security log
Backup (B)	Backup files and directories
Restore (R)	Restore files and directories
Change Time (CT)	Change the system time
Install Devices (ID)	Load and unload device drivers
Local Logon (LL)	Logon locally
Take Ownership (TO)	Take ownership of files or other objects

5.6.1.2 Resource name

The name must match exactly the name of the system level resource. Where the endpoints support the use of wildcards, the resource name for resource type *file* can be specified as /usr/*/bin/*, CONFIG??.DAT and so forth.

Note: To avoid locking everybody out or granting universal rights to key filesystems, TACF refuses to allow /*, /tmp/* or /etc/* as record names.

A UNIX directory specified as /usr/local/bin protects just that directory. To protect everything below it, specify /usr/local/bin/*.

NT directories must be qualified by a drive letter unless the system default drive is the intended target. System variables such as %SystemRoot% are not allowed. Pathname components can be delimited with either a forward slash or a back slash (/ or \).

5.6.2 Access Time Restrictions

Day-of-week and time-of-day access time restrictions are supported. These are very similar to the login time restrictions for security groups. You can specify all days, weekdays or specific days and start and stop times.

Time restrictions can be set at a global level and at the level of specific endpoints. Endpoint-specific restrictions override global restrictions.

The default is *Anyday @ Anytime*.

5.6.3 Audit Control

You can specify that an audit record is logged for failed, successful or all attempts to access a resource.

Audit attributes values can be set as follows:

ALL	All access attempts
FAILURE	Only failures
SUCCESS	Only successful events
NONE	Nothing

The effects of Audit Control set at the security resource level and at the security group level are cumulative. The default is *Failure*.

The `WARNING MODE` attribute allows access to security resources even if the access permissions are insufficient. You might use this when first defining your resources, so you can test that you have set the access permissions to an appropriate level to allow your users to do their jobs. In this mode, an event is automatically written to the audit log.

The default for warning mode is *FALSE* (checkbox unmarked). This means that access is not allowed if permission checks fail.

Audit attributes can be set at a global level and at specific endpoint levels. Endpoint-specific audit attributes override global audit attributes.

5.6.4 Default Access

What happens if you try to access a security resource that is not explicitly available through the security roles that are assigned to a security group of which you are a member? The security resource is checked for its default access permissions. This also applies to users who are not members of any security groups, who therefore do not have explicit access to any resources through security roles. See “Generic Access Permissions” on page 86 for details on how the access permissions are mapped onto the specific endpoints and “How

Access is Granted/Denied” on page 88 for details on how the access decision mechanism works.

Default access varies depending on platform and resource type. For example:

UNIX:FILE	Read
UNIX:PROGRAM	Execute
UNIX:*	Access
NT:PRINTER	Access
NT:SYSTEM	Remote Access
NT:*	Read

5.6.5 TCP Service Access

This is a valid option only when you select the TCP resource type (UNIX only).

With these attributes, you define TCP/IP services as security resources and define which hosts or IP addresses can access them. These resources are not assigned to security roles. Access to these resources is not determined by your user ID or security group to which you belong, but is resolved by checking the IP address or hostname of the source of the access request.

TCP/IP services defined in /etc/services, NIS maps or other similar locations can be defined as security resources.

Services can be specified as a name, TCP/IP port number or range of TCP/IP port numbers. TME 10 Security Management also supports dynamic port names assigned by the portmapper as specified in the /etc/rpc file.

Hosts or networks that are allowed access to TCP/IP services can be specified as follows:

- Host name, for instance rl0230b
- Fully qualified host name, for instance rl0230b.itsc.austin.ibm.com
- Wildcards on host name or fully qualified host name, for instance *.itsc.austin.ibm.com
- Network specified as two IP addresses:
  - The mask, for example 255.255.0.0
  - The match, for example 146.35.0.0

TCP Service rights can be set at a global level and at specific endpoint levels. Endpoint-specific TCP Service rights override global access rights.

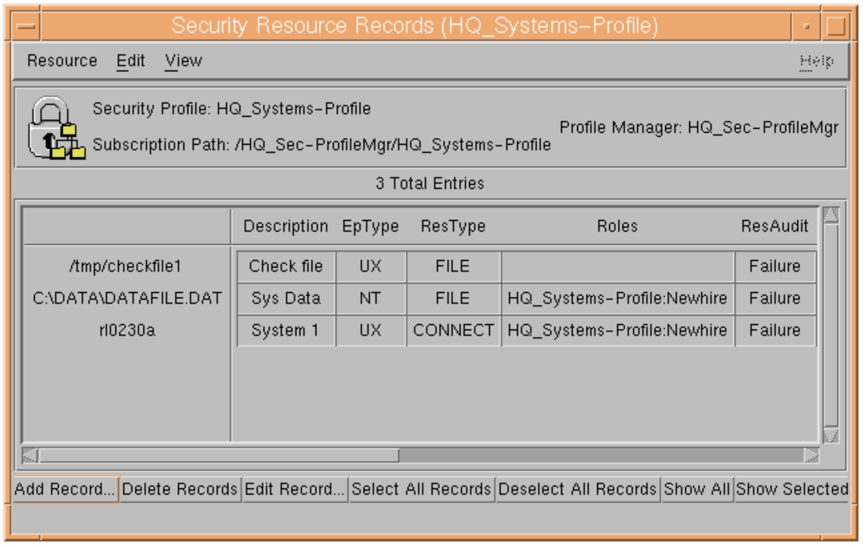
The default permission is *Access*. See “Generic Access Permissions” on page 86 for a description of how this value is mapped to the specific endpoints.

There is no default for the host/network fields.

5.6.6 Role List

You cannot assign resources to security roles from the resource interface; so this does not appear as an item in the list of actions you can perform. However, the security resource records do maintain a list of security roles to which they are assigned. This is not an editable value, but is kept to maintain integrity between the security resources and security roles.

The list of security roles can be seen in the security resource records panel under the heading Roles, as shown in Figure 22.



Security Profile: HQ_Systems-Profile
Subscription Path: /HQ_Sec-ProfileMgr/HQ_Systems-Profile
Profile Manager: HQ_Sec-ProfileMgr

	Description	EpType	ResType	Roles	ResAudit
/tmp/checkfile1	Check file	UX	FILE		Failure
C:\DATA\DATAFILE.DAT	Sys Data	NT	FILE	HQ_Systems-Profile:Newhire	Failure
rl0230a	System 1	UX	CONNECT	HQ_Systems-Profile:Newhire	Failure

Figure 22. Security Resource Records with Associated Roles

5.7 TME Security Roles

Security role records contain a list of resources and access permissions to those resources required to perform a job function. The following list of key points describe the relationship between security groups, roles and resources:

- Security resources are assigned to the security roles with the access permission required for each job function represented by the role.
- Endpoint resources from TACF/UNIX and NT that are not defined as security resources can also be assigned to security roles.
- Security groups are assigned security roles so that the users within those security groups have access to the security resources assigned to the security roles.
- Security resources may need to be accessed by several different security roles, and the access permissions to those resources can be set differently for each one.

Role definitions can be hierarchical. When you define a security role, you can specify its parent. The child security role inherits the capabilities of its parent, meaning the security resources and access levels to those resources.

The information held in the security role records includes:

Role name	A name that describes a job function
Resource list	The resources required by a job function
Resource access rights	The access permissions allowed for the resources assigned to the role

Resource type access rights	Default access rights can be assigned to each type of resource
Parent role	The name of an existing security role whose resources and access permissions are propagated into this role
Security groups	List of security groups that have the role assigned to them

The attributes (properties) associated with the information held in the records may not be applicable on every endpoint. See Table 9 on page 100 for details on how these attributes map to the specific endpoints.

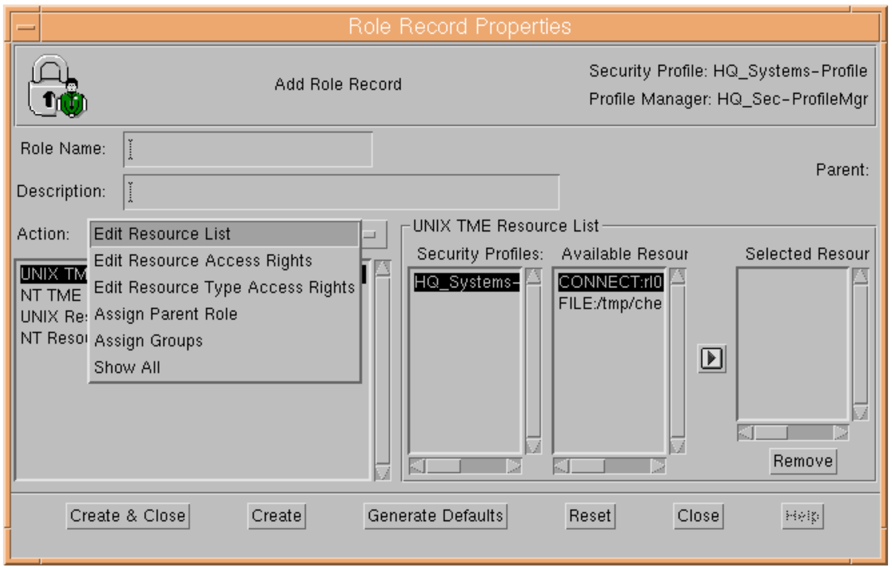


Figure 23. Dialog for Adding a Security Role Record

This figure shows the dialog panel that is displayed for adding a security role record. The information that is held in security role records is described in the following sections.

5.7.1 Security Role Name

The security role name should attempt to provide a meaningful description of the job function. This may mean that the name matches quite closely with some of the security group names to which the role will be assigned. You may want to set up guidelines for naming conventions. In our examples, we have tried to keep the security group names *people* related and the security role names *action* related. An example of a group would be Redbook_Authors and a related role would be Write_Redbook.

5.7.2 Resource List

You can select resources to be assigned to a security role from multiple sources:

- From existing security resources defined in one or more security Profiles, where the resources will be:

  <Endpoint type>:<Resource Type>:<Resource Name>

- From endpoint-specific resources (TACF/UNIX and NT), where there is not an existing security resource defined.

5.7.2.1 Resource Access Rights

Each security resource assigned to a security role has the following information in the security role record:

- The security profile name if the resource is already defined as a security resource
- Resource type, for example FILE
- Resource name, for example /usr/devt/*
- Access permissions that define access for users who are in security groups with this role assigned.
- Conditional access is supported for the UNIX endpoint entity PROGRAM type. This specifies access to a resource when executing the specified program or application.

  For example, you might specify that updates to database files can only be made through the database application binaries.

  Conditional access is not supported on NT.

A security resource may appear more than once in the list of resources assigned to a security role where each entry has unique access rights. For example, a security group may have different access to a resource when accessing it through a specific program than when accessing it directly.

See “Generic Access Permissions” on page 86 for details on how the access permissions are mapped onto the specific endpoints and “How Access is Granted/Denied” on page 88 for details on the access decision mechanism.

There are not default values for resource access rights.

5.7.3 Resource Type Access Rights

Default access rights can be set for endpoint-specific resource types. These apply when members of a security group try to access a resource where its security role does not have specific access to the resource itself.

Conditional access to resource types is supported as described for resource access rights above.

There are no defaults for resource-type access rights.

5.7.4 Parent Role

Security roles can be defined as a child of an existing security role. This will allow the child to inherit the capabilities of the parent, meaning the assigned resources and associated access permissions of the parent. The child can then be modified as follows:

- Access to new security resources can be defined that the parent does not have.
- The inherited access level to security resources can be overridden to give the child increased access.

- The inherited access level to security resources can be overridden to give the child decreased access.

See “Example 4: Child Security Roles Inheriting Parent Capabilities” on page 80.

There is no default value for this field.

5.7.5 Groups

One or more security groups can be assigned to a security role. If a security group is assigned, the record also include information about the security profile to which it belongs. This appears in the attributes table in the form *profilename:groupname*. Members of the security groups then have access capabilities to all the security resources assigned to those security roles.

As we mentioned in “TME Security Groups” on page 65, security group names can be defined at a global and endpoint level. The security role record holds information for both names (if applicable) so that the endpoint-specific group name is used for authorizing access to resources when the records are distributed. For example, the global name might be *Support_Users* and the UNIX name *support.* When the records are distributed, the endpoint-specific group name is used for authorizing access to resources.

You have already seen that security roles can also be assigned to security groups from the security group interface.

Note

When specifying an NT group name that is in a (trusted) domain outside the role’s primary domain, the name must be domain-name qualified to avoid conflicts with other groups of the same name either in the primary domain or in other trusted domains. The name is qualified using the form `"<domain name>\<group name>"`.

There is no default value for this field.

5.8 Examples

At this point, it may be helpful to discuss a few examples of the relationship between the security groups, roles and resources.

5.8.1 Example 1: User in Multiple Security groups

Within an organization, individuals can and probably will belong to more than one group. Every employee will belong to a department and perform departmental functions, but may also perform functions that require access to additional resources.

For example, suppose we have a Customer Support department that stores its files in a top-level directory called /cust_supp_dir. These files should not be generally accessible to other users. So we create a security resource for the directory with default access of *none*, then a role of *Customer_Support* that allows Read, Write and Delete access to the resource, and we assign this role to the *Support_Users* group.

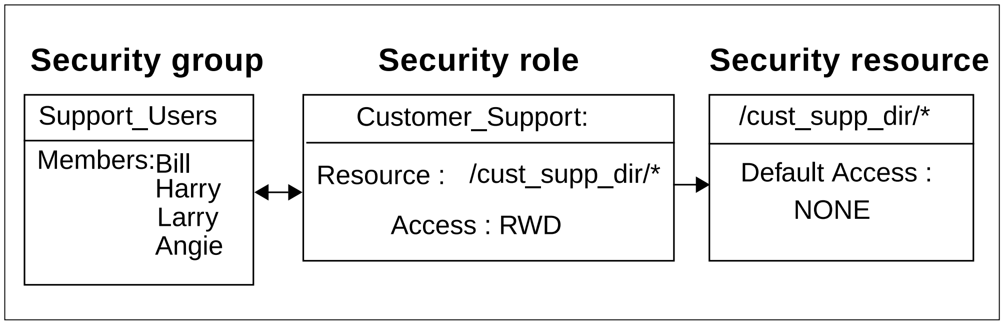


Figure 24. Support_Users Group Accessing Customer Support Database

These are the general steps that produce this result:

1. Create security resource /cust_supp_dir/*. This directory contains all the files that need to be accessed by Customer Support users; default access is set to NONE.
2. Create security role Customer_Support.
3. Add security resource /cust_supp_dir/* to security role Customer_Support with RWD access permissions.
4. Create a security group Support_Users.
5. Add all the people in the Customer Support team as members of the security group Support_Users.
6. Assign security role Customer_Support to security group Support_Users.

Angie is the manager of the Customer Support team; so she needs access to additional resources that are available only to managers.

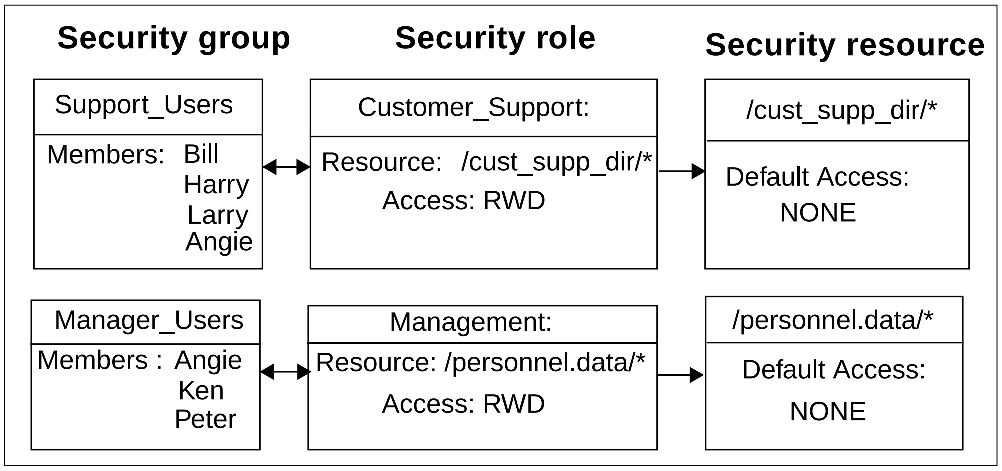


Figure 25. Manager_Users Group Accessing Personnel Data

1. Create security resource /personnel.data/* with default access set to NONE.
2. Create security role Management.

3. Add security resource /personnel.data/* to security role Management with access permissions of RWD.
4. Create security group Manager_Users.
5. Assign security role Management to security group Manager_Users.

When Angie signs on, she belongs to security groups Support_Users and Manager_Users and will therefore be able to access all the resources she needs.

If Angie changes her job, where she is no longer a manager, her access to the personnel data files needs to be removed. This can be achieved with a single action:

- Remove Angie as a member of the security group Manager_Users.

This automatically removes her access to all the security resources accessed through the security role Management.

Without groups, a change in job would mean changing access to many different resources. Even when using groups in a UNIX fashion, a change in status or job might mean the changing of many different groups to ensure the correct access for the user. Using security groups and security roles allows us to give or deny access to multiple resources in a single action. The security group could, of course, have multiple roles assigned to it. Removing the user from the group removes all of those roles in the same, single action.

You might prefer to avoid memberships of multiple groups by making greater use of roles. Angie might be a member of one group, say Support_Managers, which has two roles assigned to it, Customer_Support and Management. Then if Angie were to leave the company, she would only need to be removed from the Support_Managers group to remove all accesses. This is similar to the next example where multiple roles are employed. In the real world, the power of this use of groups and roles comes into its own when we are talking about large numbers of roles and resources.

For a discussion on role-based and group-based approaches, see Section 2.3.2, “Implementing an Organizational Approach” on page 14.

5.8.2 Example 2: Security Group with Multiple Roles

As we mentioned earlier, security groups can have, and will most likely need, more than one security role. Here is an example of how this works:

Suppose we have a security group called *Redbook_Authors* whose members are responsible for writing this redbook. The group is assigned the security role *Write_Redbook*, which gives it access to /tme/redbook/* with Read and Write permissions. The users writing the redbook are also required to add information about good places to see in Austin in resource /austin_hotspots. This resource is available to everybody to read, but group Redbook_Authors needs an additional role that gives Read and Write access to the resource.

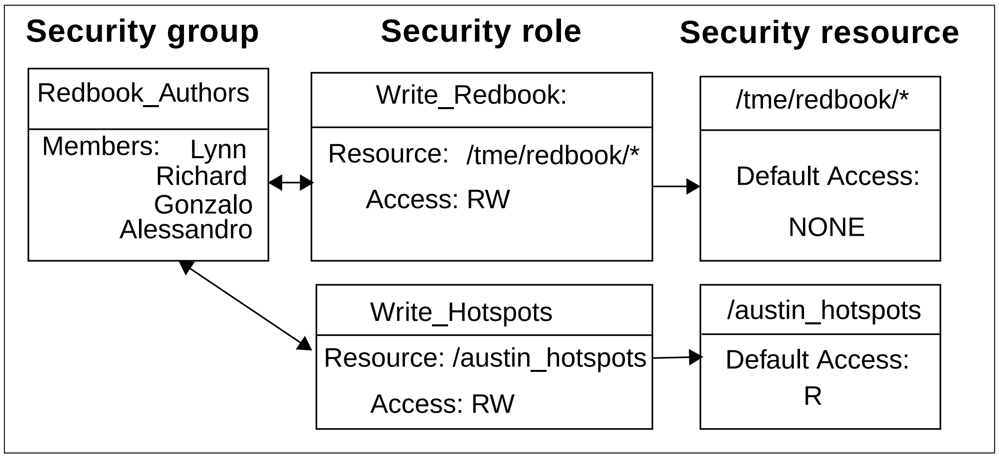


Figure 26. Security Groups with Multiple Security Roles Assigned

These are the general steps that produce this result:

1. Create security resource /tme/redbook/* with default access None. This contains all the chapters of this redbook.
2. Create security resource /austin_hotspots with default access Read (R). This contains information on all the good places to visit in Austin.
3. Create security role Write_Redbook.
4. Create security role Write_Hotspots.
5. Add security resource /tme/redbook/* to security role Write_Redbook with access permissions of RW.
6. Add security resource /austin_hotspots to security role Write_Hotspots with access permissions of RW.
7. Create security group Redbook_Authors.
8. Assign security role Write_Redbook to security group Redbook_Authors.
9. Assign security role Write_Hotspots to security group Redbook_Authors.

5.8.3 Example 3: Multiple Security Roles Accessing Same Security Resources

We have a security group Redbook_Authors as shown in Figure 26 on page 79, but we want to allow some other users to have Read access so they can review the book but not change it. So we create a new security role that has access to the security resource /tme/redbook/* with Read access and assign the new role to security group Redbook_Reviewers. Users in this security group now have Read access to the resource.

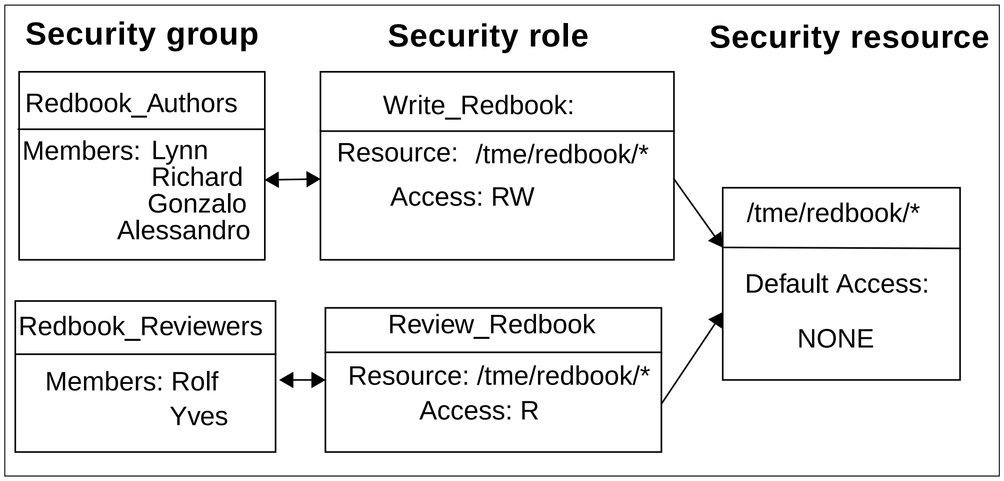


Figure 27. Multiple Security Groups Accessing Security Resources

These are the general steps that produce this result:

1. Create security role Review_Redbook with access permissions to security resource /tme/redbook/* set to Read (R).
2. Create security group Redbook_Reviewers and add Rolf and Yves as members of the group.
3. Assign security resource /tme/redbook/* to role Redbook_Reviewers (with Read access).
4. Assign security role Review_Redbook to security group Redbook_Reviewers.

5.8.4 Example 4: Child Security Roles Inheriting Parent Capabilities

Here we have a group of developers in security group *Developers* who have Read, Write and Delete access to source libraries and documentation in /usr/devt/src/* and /usr/devt/docs/* through the security role *Development*. The default access to the source code and documentation is None.

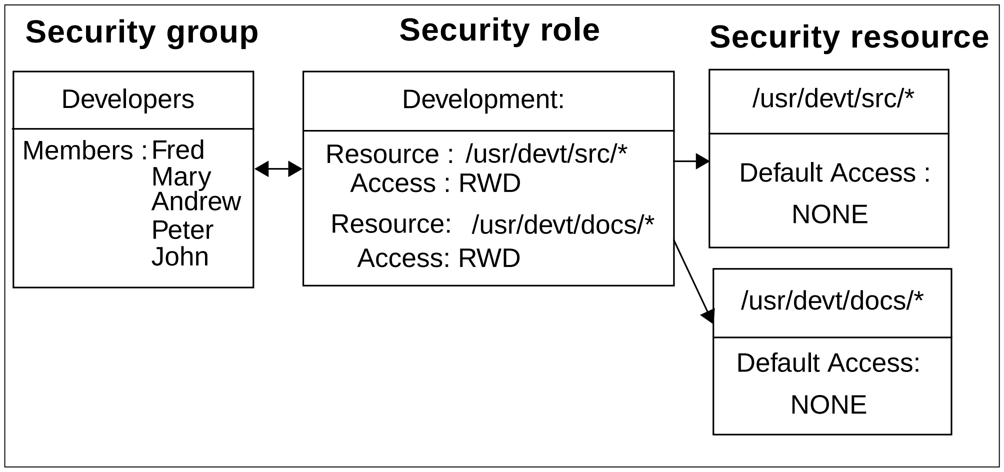


Figure 28. Security Group Accessing Security Resources via Security Role

These are the general steps that produce this result:

1. Create security resources called /usr/devt/src/* and /usr/devt/docs/* with default access None. These directories contain all the source code and documentation that the developers need.
2. Create a security role called Development.
3. Add the security resources to security role Development with RWD access.
4. Create a security group called Developers.
5. Add all the user IDs of the people in the development group to the security group Developers.
6. Assign the security role Development to the security group Developers.

New contract programmers Jenny and Rick are employed to develop some confidential code that the rest of the development group will not be able to access, but the new employees are allowed only Read access to the standard development libraries.

So we create a new security role Development_Conf specifying Development as its parent so that the new role inherits its resources and access permissions. We create a new security group for the contractors and assign the new role to the group, changing the access permissions to its resources to Read-Only. We then assign an additional resource /usr/devt/conf/src/* to the new role with Read, Write and Delete access.

When Jenny and Rick sign on, they now have access to the confidential resources through security role Development_Conf. They have Read-Only access to the standard development resources because the access permissions on the inherited capabilities from security role Development have been changed. Users in the security group Developers can still access their source code and documentation with Read, Write and Eelete access, but they cannot access /usr/devt/conf/src/* because that has been defined with default access of None.

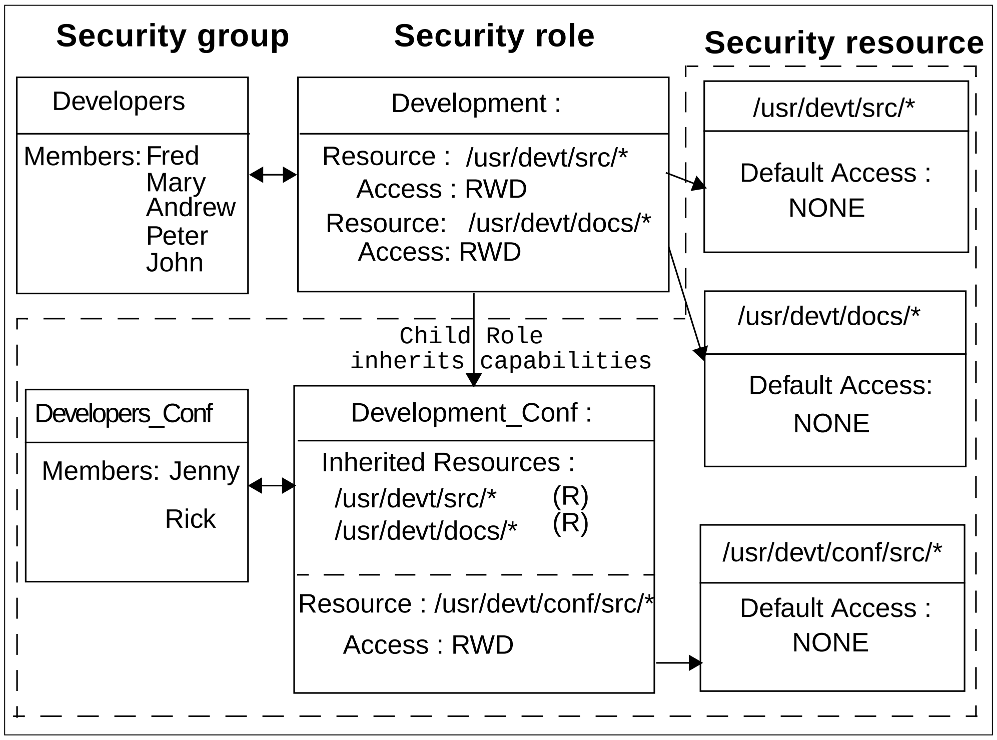


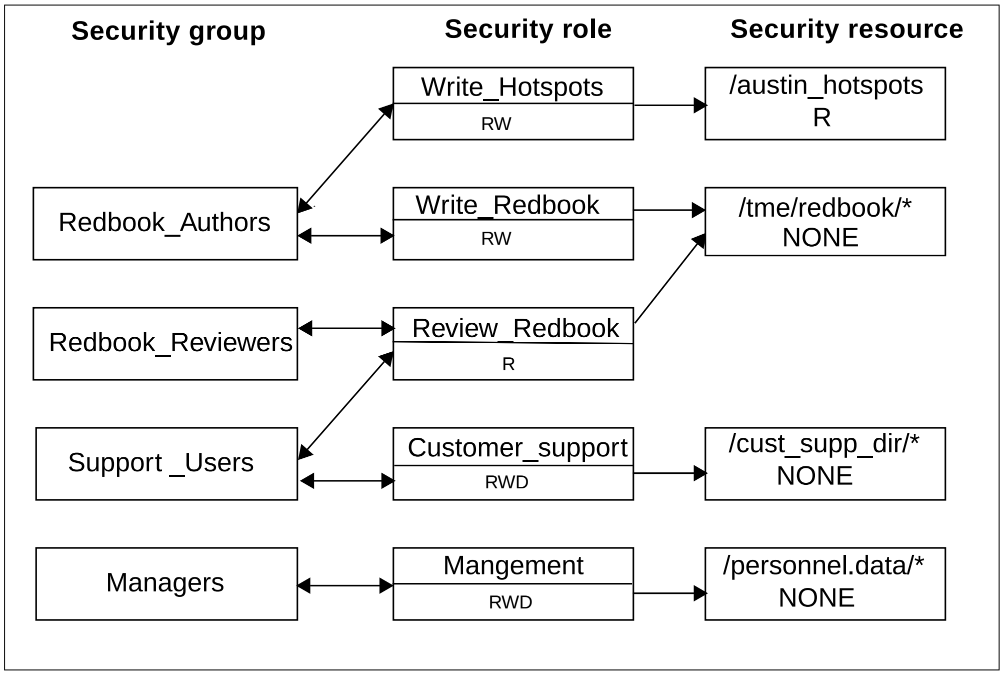
Figure 29. Child Role Inherits Capabilities of Parent

1. Create a security resource called /usr/devt/conf/src/* with default access of None.
2. Create a security role called Development_Conf. Specify security role Development as the parent of this security role as you create it, so that the new role will inherit the capabilities of the parent.
3. Change the inherited permissions on security resources /usr/devt/src/* and /usr/devt/docs/* to Read-Only.
4. Assign the security resource called /usr/devt/conf/src/* to the security role Development_Conf with RWD access.
5. Create a security group called Developers_Conf.
6. Assign users Jenny and Rick as members of security group Developers_Conf.
7. Assign security role Development_Conf to security group Developers_Conf.

5.8.5 Example 5: Linking Some of the Examples

We have shown some examples related to each component of the role-based implementation. Let’s put some of those together.

We would like the security group Support_Users to have Read access to the redbooks so they can use the information as an additional source for answering customer questions. We already have role Review_Redbooks that can access /tme/redbooks/* with Read access; so we assign that role to the security group Support_Users.



5.9 TME System Policy

TME 10 Security Management provides a mechanism for system-wide definition of password policies, audit polices, login policies, and default access to resource types. This means that policies can be defined once, distributed to all systems and applied consistently to all users. Figure 30 shows a conceptual view of the relationship between the system independent definition of policies and the distribution of those policies to the multiple endpoint types with differing native security systems.

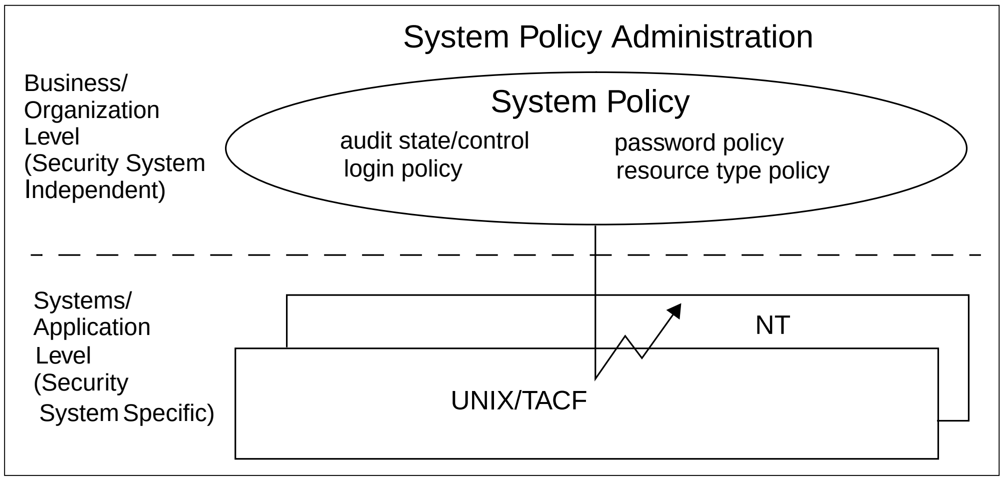

System Policy Administration
Business/ Organization Level (Security System Independent)
System Policy
audit state/control
login policy
password policy
resource type policy
Systems/ Application Level (Security System Specific)
NT
UNIX/TACF

Figure 30. Conceptual View of System Policy Administration

The systemwide policies are defined in TME 10 Security Management system policy records which are created in a security profile. For security groups, roles and resources in resource access administration, there can be many records defined. However, in security system policy, there will be very few system policy records–perhaps only one. By definition, the system policy record contains information that you want to be applied to all systems. However, within each record, you can specify the system policy attributes at a global level and at the level of the endpoint type. In this way, you can define different policies for each endpoint type.

Note: TME 10 User Administration can also define user-specific security polices, for example, the minimum and maximum times allowed between changing passwords. Where these values are defined for users in user profiles, they override the TME 10 Security Management system policies.

Where audit attributes and resource access permissions are defined in security group, role or resource records, they override the system policy values.

System policy records contain the following information:

- Audit control attributes
- Login policies
- Password policies
- Resource type access policies

The attributes (properties) associated with the information held in the records may not be applicable on every endpoint. See Table 10 on page 101 for details on how these attributes map to the specific endpoints.

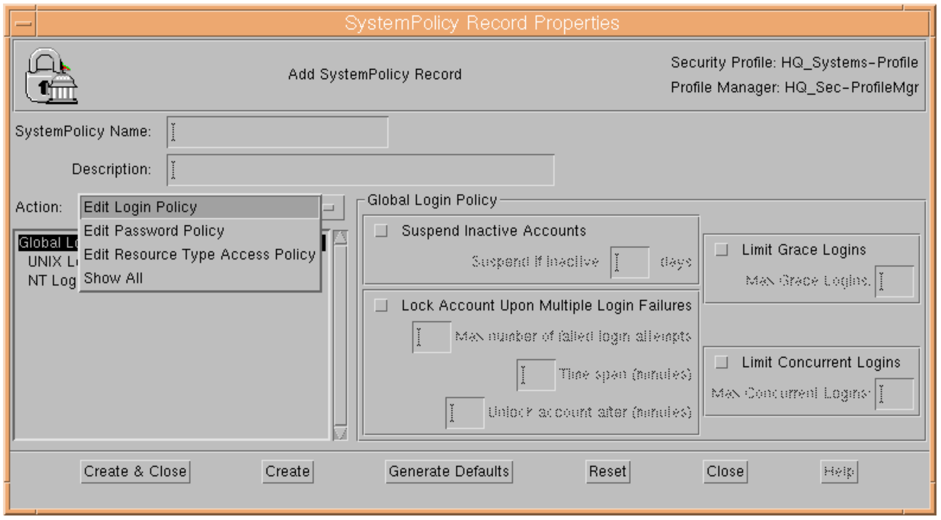


Figure 31. Dialog for Adding a System Policy Record

This figure shows the dialog panel that is displayed for adding a security system policy record. The information that is held in security system policy records is described in the following sections.

5.9.1 Login Policy

You can specify the following values (defaults are in parentheses):

- *Suspend inactive accounts* - suspend user's login after specified number of days in which the user does not log in (never)

- *Lock account upon multiple login failures* - enable or disable whether users are locked based on the number and frequency of failed login attempts (never). If enabled:
  - *Lockout after* - lock after specified number of failed logins
  - *Lockout harmless count* - number of minutes during which failed login attempts will be counted, the failed count being reset at each harmless count interval
  - *Lock duration* - how long the user account should remain locked
- *Limit grace logins* - limit the number of logins allowed after user's password has expired (6)
- *Limit concurrent logins* - limit the number of concurrent logins for user (unlimited)

You can specify how login attempts are audited:

- *Login successes/failures* - audit success and/or failure of system-wide login of users (none)
- *Resource access successes/failures* - audit success and/or failure of system-wide resources by resource type (none)

Login policies can be set at a global level and at the level of specific endpoints. Endpoint-specific restrictions override global restrictions.

5.9.2 Password Policy

You can enable or disable password checking. If enabled, we set the following (defaults are in parentheses):

- Minimum days between password changes (Now)
- Maximum days between password changes (42)
- Minimum password length (5)
- Minimum alphabetic characters (1)
- Minimum alphanumeric characters (1)
- Minimum numeric characters (1)
- Minimum uppercase characters (1)
- Minimum lowercase characters (1)
- Maximum repeated characters (2)
- Minimum special characters (1)
- Password history depth - number of previous passwords stored (none)

You can also allow the Administrator to change a user's password if it expires. This option is set to false by default.

Password policies can be set at a global level and at the level of specific endpoints. Endpoint-specific restrictions override global restrictions.

5.9.3 Resource Type Access Policy

For each endpoint type, you can do the following:

- Enable or disable resource type access checking for each resource type.

  The default is *Enabled*.
- Specify the default access for each resource type.
- Specify the default audit values for each resource type.

The defaults are set to *Read* or *Access* depending on the resource type.

See "Generic Access Permissions" on page 86 for details on how the access permissions are mapped onto the specific endpoints and "How Access is Granted/Denied" on page 88 for details on the access decision mechanism.

5.10 Guided Tour through Access Permissions

TME 10 Security Management provides a set of system-independent, generic access permissions. These are described in the next section.

We will then look at the different levels in which you can define access to resources and resource types and show you some examples of how decisions are made on whether access should be allowed or denied.

5.10.1 Generic Access Permissions

Access permission names and values vary between operating systems. For example, the Read access permission for a file is *r* on UNIX, but it would be *READ* once RACF is supported in the product. TME 10 Security Management provides a set of generic access permissions that are mapped onto the system-specific names when the records are distributed.

The set of access permissions supported varies depending on the resource type. The dialog panels associated with different resource types display only the values that are valid for that resource type.

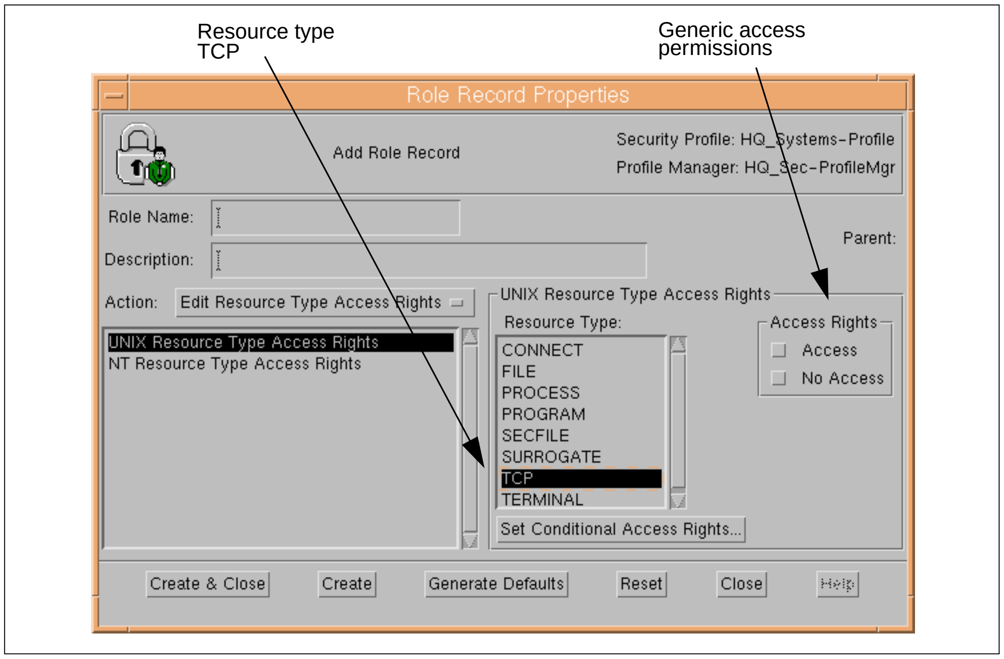

Resource type TCP
Generic access permissions
Role Record Properties
Add Role Record
Security Profile: HQ_Systems-Profile
Profile Manager: HQ_Sec-ProfileMgr
Role Name:
Description:
Parent:
Action: Edit Resource Type Access Rights
UNIX Resource Type Access Rights
NT Resource Type Access Rights
UNIX Resource Type Access Rights
Resource Type:
CONNECT
FILE
PROCESS
PROGRAM
SECFILE
SURROGATE
TCP
TERMINAL
Access Rights
Access
No Access
Set Conditional Access Rights...
Create & Close
Create
Generate Defaults
Reset
Close
Help

Figure 32. Generic Access Permissions for Resource Type TCP in Role Records

The following tables show generic access permissions that are available on each resource type. You should experiment with the different options to ensure you have the required levels of access. This experimenting is where the warning mode is very useful.

5.10.1.1 Files and Directories

Files and directories have the following generic access permissions:

Table 3. Generic Access Permissions for Resource Type File

Resource Type: FILE	Access Permissions
Read (R)	User can open and browse files and display files and subdirectories in a directory
Write (W)	User can modify files, create subdirectories and add files to a directory
Execute (X)	User can execute an executable program
Delete (D)	User can delete a file
Update (U)	Combines Read, Write, Execute and Delete and allows directory to be deleted
Change Ownership (O)	User can change owner
Change Permissions (P)	User can change permissions
Full Control (F)	User has full control of files and directories, including permissions and owner
No Access (N)	Denies all access to files and directories
NOTE: A Create permission exists in TACF, but is not implemented from the GUI in the first release of TME 10 Security Management. To create a file in a wildcard-protected directory, the user would need the Control (C) access permission.	

5.10.1.2 Printers

Printers (NT only) have the following generic access permissions:

Table 4. Generic Access Permissions for Resource Type Printer

Resource Type: PRINTER	Access Permissions
Access (A)	User can submit job to a printer
Full Control (F)	User has full control of printer, including pausing and resuming jobs, changing priorities and changing printer permissions
No Access (N)	Denies all access to a printer

5.10.1.3 General Resource Types

All other resource types have the following generic access permissions:

Table 5. Generic Access Permissions for General Resource Types

Resource Type: GENERAL	Access Permissions
Access (A)	User can access or use resource
No Access (N)	Denies all to resource

5.10.1.4 Mapping Generic Access to Endpoints

When the security records are distributed, the generic access permissions are mapped onto the specific system-level values as follows:

Table 6. Mapping Generic Access to Endpoint Specific Values

	Read	Write	Execute	Delete	Update	Change Owner	Change Perms	Control	Access	No Access
TACF										
FILE	Read	Write	Execute	Delete	Read Write Execute	chown	chmod	chown chmod utimes sec Update		None
PROCESS									Read	None
PROGRAM									Read	None
SECFILE									Read	None
CONNECT									Read	None
TCP									Read	None
TERMINAL									Read	None
SURROGATE									Read	None
NT										
FILE	R	W	X	D	Change RWXD	O	P	Full control		No access
DIRECTORY	R	Add			Change RWD	O	P	Full control		No access
SHARE	R				Change RWXD			Full control		No access
PRINTER								Full control		No access

5.10.2 How Access is Granted/Denied

You have seen that we can define access permissions in several places:

- Security resource records
- Security role records for access to the resource
- Security role records for default access to resource types
- System policy records for default access to resource types

In this section, we look at some examples of access being set through these four different record types. We are looking specifically here at the TACF implementation. The Windows NT permission granting process is well described in other books. See Section 3.1.2.5, “Access Control Lists and Security Descriptors” on page 26, for a brief overview of the NT process.

There are several key points to note about the TACF implementation before we look at some details:

1. TACF checks authorization to resources based on the original user ID. So if you login as fred and then su to root, you will still have access only to the resources to which fred is permitted. You will not gain access to root’s resources.
2. Resource type *file* is treated differently from the other resource types. The default access level for resource type *file*, both in the security role and system policy records, is set to Read. However, consider what would happen where a user is accessing a file that has not been defined as a security resource and does not have a record in the TACF database:

  - If the user is a member of a security group, the default resource type access rights in the group's assigned roles will be checked
  - If the user is not a member of security group, the default resource type access rights in the system policy record will be checked

  If the default access is set to Read, then this is the maximum level of access the user will get to files on the system except for those explicitly available through the user's (group's) roles. The majority of files on a UNIX system will probably not be defined as security resources. A default access of Read would create a risk of TACF failing application attempts to create temporary files or open shared libraries during program load. We could specify a default of All, but that makes the default fairly meaningless.

  So what actually happens is that the default resource type access rights for file are bypassed, and access checks are done at the UNIX level. Remember though that this does not apply to files that are specifically defined as security resources; their default access or access defined through a role will apply before the UNIX-level access is checked.

  TACF does make use of the default you set for resource type file in some circumstances. In TACF itself, although not directly accessible through the TME 10 Security Management GUI, there is a group called _restricted. Users added as members of this group will have all their accesses to files checked.

3. Commands and executable programs that are not setuid or setgid, and are not specifically defined as security resources, are allowed by TACF to execute. If the program is setuid or setgid, TACF checks the default access allowed in the PROGRAM class.
4. TACF can deny you access to resources, but it cannot explicitly allow you access. Think of TACF as series of tests you have to pass before you are passed back to UNIX with TACF setting the maximum level of access that you will eventually be allowed.

   For example, in TACF, a user might have Read access to a file, but unless the UNIX permissions also allow Read access for the user, access is denied. If TACF says you can Read and UNIX says you can Read and Write, the TACF Read-Only access prevails.
5. Users can belong to multiple security groups, and groups can have multiple roles assigned to them. Roles may have different access permissions to the same resources and may set different defaults for resource type access. So when a user wants to access a resource, TACF looks at the resources and file permissions of all the roles allocated to all the groups in which the user is defined. If there are different levels of access to the same resource, either though the specific access permissions or through the default resource type access, then by default TACF applies cumulative levels of access.

Here is a simplified view of the steps that are followed to determine whether or not a user can access a resource:

1. When the resource is not defined as a security resource and there is not an existing record for the resource in TACF:
   - Is resource type access checking enabled for this resource type?
     – If NO, then pass request to UNIX
   - Is the resource type a FILE? – If YES, pass request to UNIX
   - Is user defined in TACF?

    – If YES, then is there a resource type access defined in users roles?
    – If resource type access is none, fail request
    – If resource type access is Read or above, pass request to UNIX
  - is there a system policy defined?
    – If NO, then pass request to UNIX
    – If YES, if default access is NONE, then fail request
    – if YES, if default access is Read or more, pass request to UNIX

2. Where the resource is defined in TACF.

  - Is resource type access checking enabled for this resource type?
    – If NO, then pass request to UNIX
  - Is there specific access to resource via roles assigned to user's group
    – If specific access in None, fail request
    – If specific access is Read or above, pass request to UNIX
  - If there is no specific access via roles, check default access of resource
    – If default access is NONE, fail request
    – If default access is Read or above, pass request to UNIX

In addition to these steps, access time restrictions and conditional access restrictions are checked.

Let's look at some specific examples.

Example 1:

User fred is not a member of any security groups and wants to access his own files in /home/fred. None of the files below /home/fred are defined as security resources. So from the checklist above, we can see that the request is passed directly to UNIX. User fred owns the files in his home directory; so UNIX allows access.

Example 2:

User fred is not a member of any security groups and wants to Telnet to a system called rl0203c.itsc.austin.ibm.com. This system is not defined as a CONNECT resource.

If resource type checking is enabled for resource type CONNECT, and there is a system policy record defined, then the default level of access to that resource type is applied.

Example 3:

User fred is not a member of any security groups and wants to Telnet to a system called rl0230a.itsc.austin.ibm.com. This system is defined as a CONNECT resource with a default access of none.

User fred is denied access by the resource's default access level.

Example 4:

User lynn is a member of security group Redbook_Authors. Security role Write_Redbook has Read and Write access to security resource /tme/redbook/* which has default access of None and Read access to rl0230a.itsc.austin.ibm.com. The role is assigned to the security group Redbook_Authors.

TACF allows user lynn Read and Write access to /tme/redbook/* through the role access permissions and passes the request on for UNIX to check its permissions. User lynn can also Telnet to rl0230a.itsc.austin.ibm.com through the role permissions.

Example 5:

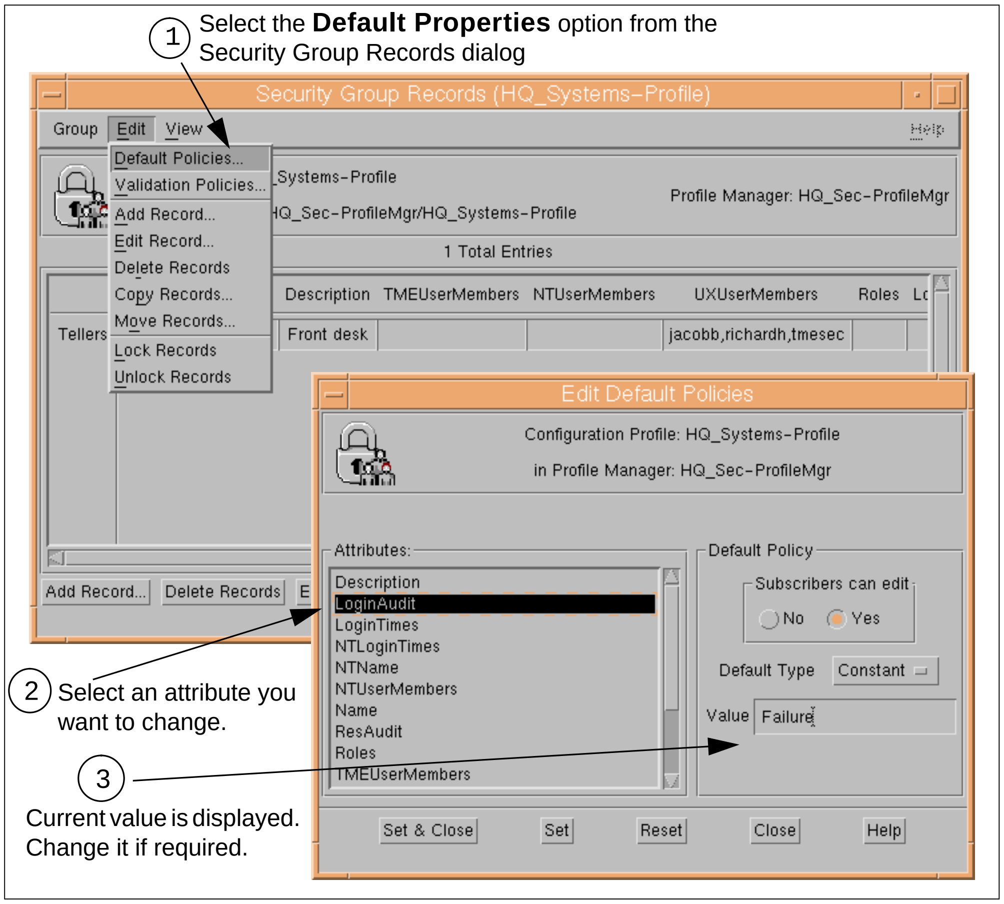
User lynn is added as a member to a second security group Customer_Support which has Read access to /tme/redbook/* through security role Review_Redbook. User lynn is now only able to read /tme/redbook/*. The access permissions in TACF shows that security group Redbook_Authors has Read and Write access and Review_Redbook has read access, but because lynn is a member of both groups, then by default, the cumulative access level is applied (Read and Write in this case).

5.11 Default and Validation Policies

Default policies contain default values that are applied if specific values are not supplied when a new record is created in a profile.

Validation policies contain valid values that are allowed for an attribute when a record is created or modified. Validation policy can be disabled. This can be useful when populating profiles from existing system-level information, where some of the records will have values that will not pass the validation test, but you still want to put them into the profile. In these cases, you can then change the record to have a valid value, re-enable validation and redistribute the profile to change the system-level information at the endpoint.

Security group, security role, security resource, and system policy records all have default and validation policies, although not every attribute within the records has a default value. For example, it would not be possible to set default values for user IDs who are to be added as members of a security group.


1 Select the **Default Properties** option from the Security Group Records dialog

2 Select an attribute you want to change.

3 Current value is displayed. Change it if required.

Profile Manager: HQ_Sec-ProfileMgr

1 Total Entries

Description	TMEUserMembers	NTUserMembers	UXUserMembers	Roles
Front desk			jacobb,richardh,tmesec	

Configuration Profile: HQ_Systems-Profile

in Profile Manager: HQ_Sec-ProfileMgr

Default Policy

Subscribers can edit: No [✓ Yes]

Default Type: Constant

Value: Failure

Figure 33. Viewing and Changing Default Policies

Figure 33 shows the steps for viewing and changing default policies. Many default and validation policies have the value set as a *constant,* as in this example. The default type can be constant or script. If the default type is script, you can edit the script from this dialog panel and also change the arguments that are passed to the script.

You should review the default and validation policies for all record types before creating new records to ensure consistency through all the records in the security profile. Remember, though, that you can create different security profiles if you want to apply different defaults to some of your systems.

Full details on changing default and validation policies can be found in the *TME 10 Security Management User Guide.* See also the discussion in Section 4.3.4, “Policy and Policy Regions” on page 49.

5.12 Populating the Security Profiles

After setting your default and validation policies, you may decide to populate the security profile with existing system-level information. Populating the security profiles is supported at several levels:

- Security groups, resources and system policy records can be populated from all system-level groups, resources and system policies of selected endpoint systems. This option is selected from the security profile panel:

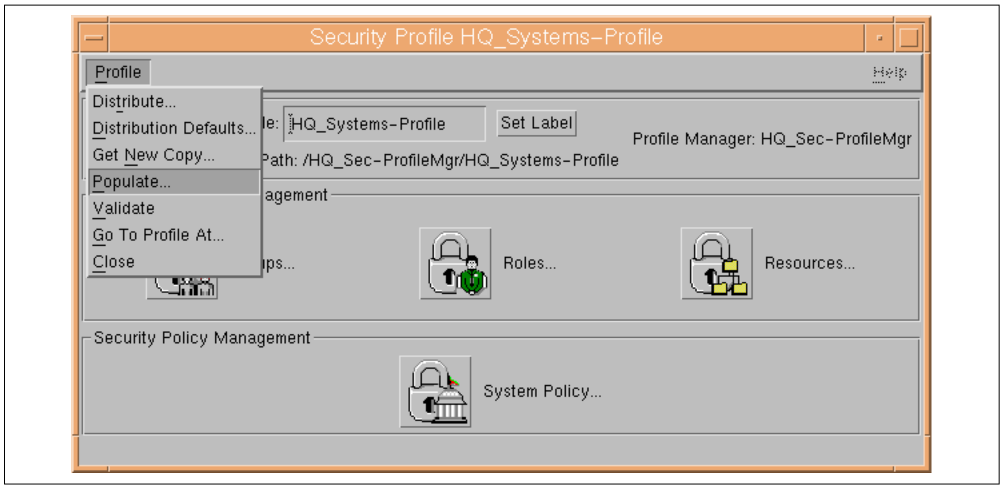


Figure 34. Populate All Record Types in Security Profile

- Population of security records can be done within the context of the security record type for security groups, resources, roles, and system policies.

Figure 35 shows the dialog panels used for populating the system policy records.

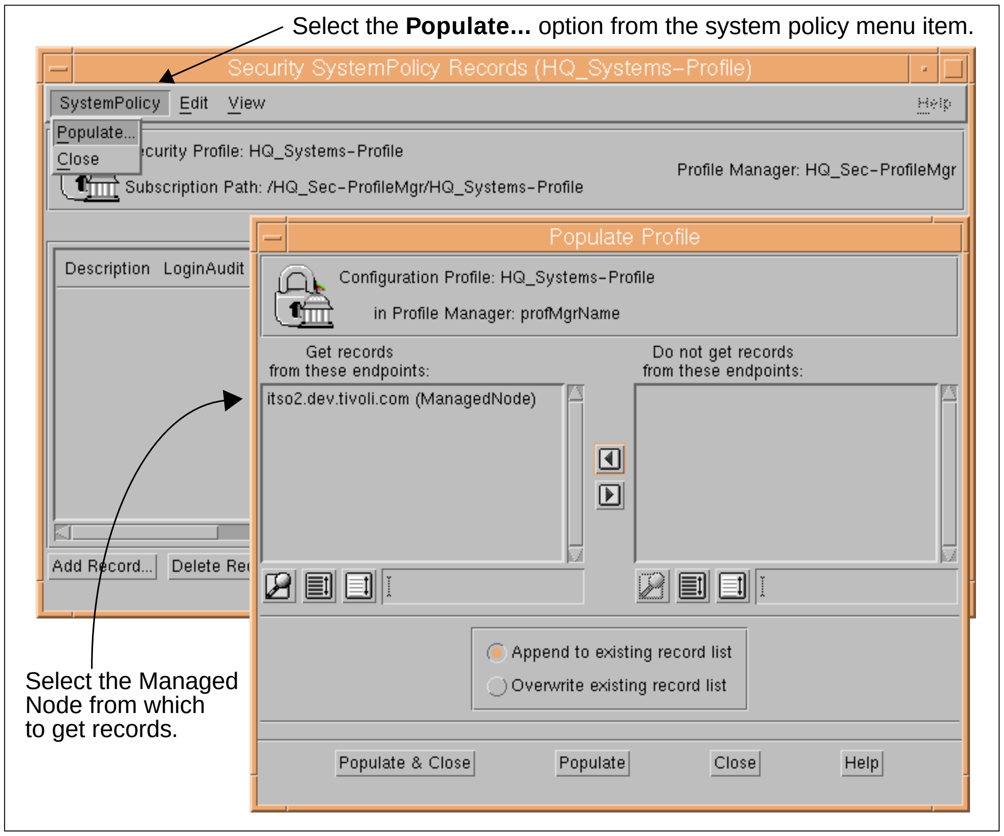

Select the **Populate...** option from the system policy menu item.
Security SystemPolicy Records (HQ_Systems-Profile)
SystemPolicy
Edit
View
Help
Populate...
Close
curity Profile: HQ_Systems-Profile
Subscription Path: /HQ_Sec-ProfileMgr/HQ_Systems-Profile
Profile Manager: HQ_Sec-ProfileMgr
Description
LoginAudit
Add Record...
Delete Re
Populate Profile
Configuration Profile: HQ_Systems-Profile
in Profile Manager: profMgrName
Get records from these endpoints:
itso2.dev.tivoli.com (ManagedNode)
Do not get records from these endpoints:
Append to existing record list
Overwrite existing record list
Populate & Close
Populate
Close
Help
Select the Managed Node from which to get records.

Figure 35. Populating Security Records

You specify one or more managed nodes from which you want to collect information for populating your security records. You have the option when populating security group and resource records to specify:

- Populate all records of the selected record type
- Populate all records with names matching a specified regular expression
- Populate a list of specified record names

Population of security role records can only be done from roles that already exist in TACF or from local groups from NT.

The following sections show what happens when profile records are populated.

5.12.1 Security Group Records

Security group records are populated depending on the type of operating system as follows:

- UNIX

  The source of the information imported from UNIX is the TACF database. If you are starting from scratch, the TACF database only contains group information for a small number of default groups; so this may be of limited use.

  It creates a security group record for each TACF group. The user members of the TACF group are added to the security group member list.

- NT

  A security group record is created for each NT global group. The NT user account members of global groups are added to the security group member list.

  A security group called *NT-Domain Admins* from the NT Domain Admins global group is also created. By default, it is assigned a security role called NT Administrator. Users can then be given NT domain administrative rights by adding them to the security group NT-Domain Admins. Note that this grants NT Administrator rights for that user to any subscribed domain controller to which that security profile is distributed.

5.12.2 Security Resource Records

Security resource records are populated depending on the type of operating system as follows:

- UNIX

  Creates a security resource record for each TACF resource, with the security resource class based on the TACF resource class and the security resource attributes based on the TACF resource’s security attributes.

  This may only prove a useful option if you already have a TACF database fairly well populated with resources. If you are installing from scratch, the TACF database only contains a few resources that are set up automatically at install time, such as /etc/security/passwd.

- NT

  Creates a security resource for each NT resource assigned to an NT Local Group, mapping the resource type and setting the security resource name the same as the NT resource.

It also creates security resources with resource type *NT Rights* for all NT Rights. However, resource type NT Rights is not an editable type when creating or editing resources from the GUI or the CLI. Security resources with resource type NT Rights are visible wherever security resources are normally seen, for example, from the security role resource list selection. But they cannot be edited and there are no permissions to be set for them.

5.12.3 Security Role Records

Security role records cannot generally be populated in the same way as security group and security resource records. NT is an exception to this and is described in more detail below. However, it is possible to selectively populate security roles as follows:

- UNIX

  The populate option from UNIX creates a security role in a profile from a role previously created in TACF or distributed from other profiles to TACF.

- NT

  Population creates a security role with the same name as each NT Local Group. The default set of security roles discovered, based on the default set of NT Local Groups, are:

  - NT Administrator
  - NT Account Operator
  - NT Server Operator
  - NT Print Operator
  - NT Backup Operator
  - NT Power Users (NT workstation only)
  - NT Users
  - NT Guests

  For each NT Global group assigned to the NT local group, add the matching security group to the new security role’s list of assigned groups.

  For each NT resource (for example FILE or NT RIGHTS) defined to the local group, it adds the resource to the security role’s NT resource access list along with its associated permissions.

5.12.4 System Policy records

A system policy record is created from the attributes for a specified endpoint with its associated security policy attributes. The system policy records are given the name *syspol*. Typically, you may want to populate the system policy from one machine and use that to distribute the policy to many others. You are only likely to want one system policy record in an one profile—there is not point in distributing multiple system policies to the same machines.

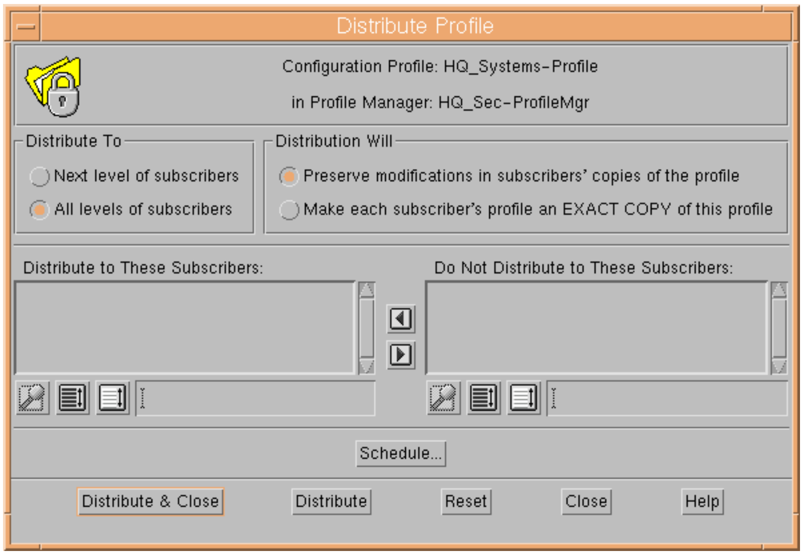
5.13 Distributing the Security Profiles

Distribution is handled by the TME 10 Framework services. We do not reproduce here all the information available on distribution that is already available in the *TME 10 Framework Planning and Installation Guide* and in other redbooks. In future releases, TME 10 Security Management will support record-level subscription (see the Futures chapter, Section 8.4.1, “Record-Level Subscription” on page 186).

In the first release, subscription is supported only at the Profile Manager level in TME 10 Security Management. Profile manager subscription determines which endpoints and other profile managers receive the contents of the security profile.

5.13.1 Distribution Options

Figure 36 shows the dialog that is displayed when you select the **Distribute...** option from the security profile dialog.



Distribute Profile
Configuration Profile: HQ_Systems-Profile
in Profile Manager: HQ_Sec-ProfileMgr
Distribute To
Next level of subscribers
All levels of subscribers
Distribution Will
Preserve modifications in subscribers' copies of the profile
Make each subscriber's profile an EXACT COPY of this profile
Distribute to These Subscribers:
Do Not Distribute to These Subscribers:
Schedule...
Distribute & Close
Distribute
Reset
Close
Help

Figure 36. Distributing a Security Profile

There are two modes in which the records can be applied at the endpoints:

- Preserve modifications in subscribers' copy of the profile
- Make subscribers' copy an EXACT COPY of this profile

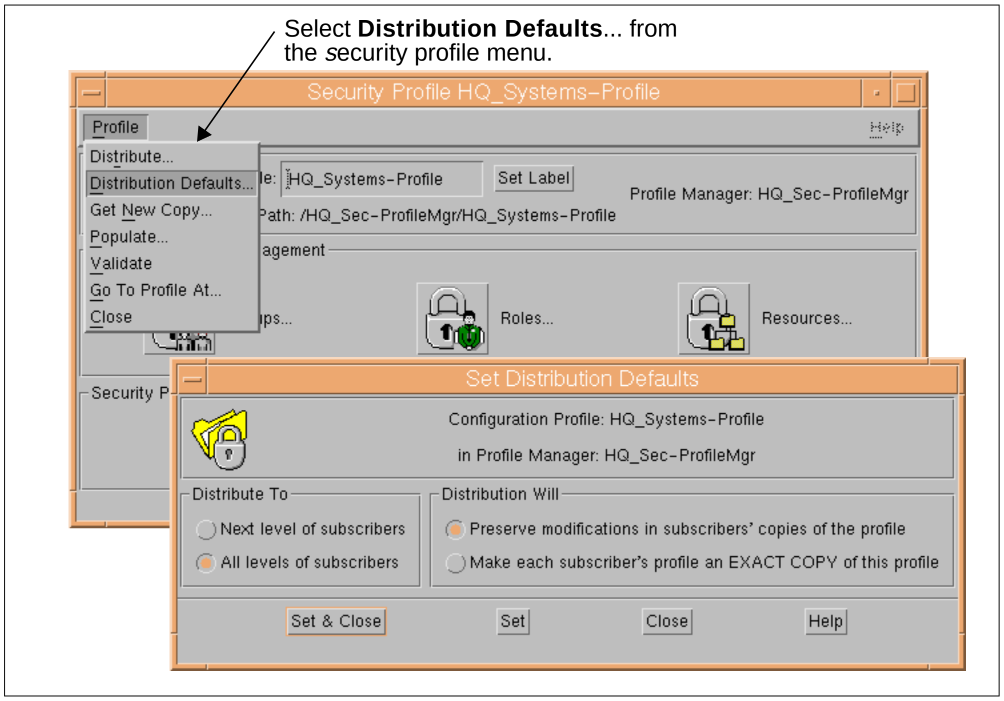
EXACT COPY Alert!

We strongly recommend that you **do not** use the EXACT COPY option.

This option updates the endpoint with the records you are distributing, but it also removes anything else it finds on the target system that does not exist in the security profile you have distributed.

So, unless your security profile contains ALL the security information that you need for the target system, **do not** use EXACT COPY.

To ensure that you do not distribute the security profiles with the EXACT COPY option, you should modify the Distribution Defaults for the security records.



le: HQ_Systems–Profile

Profile Manager: HQ_Sec–ProfileMgr

Path: /HQ_Sec–ProfileMgr/HQ_Systems–Profile

Configuration Profile: HQ_Systems–Profile

in Profile Manager: HQ_Sec–ProfileMgr

Distribute To: Next level of subscribers [✓ All levels of subscribers]

Distribution Will: [✓ Preserve modifications in subscribers' copies of the profile] Make each subscriber's profile an EXACT COPY of this profile

Figure 37. Setting Distribution Defaults

Selecting the **Distribute...** option from the security profile dialog will distribute all security record types. This is unlike the **Populate...** option, where you also have the choice of populating individual record types.

For Windows NT, exact-copy distribution is initially supported for groups and roles only. Also, default NT global and local groups such as Domain Admins or Administrators are reserved group names and are exempt from exact-copy distributions and remove-operation distributions.

5.13.2 Mapping Distributed Records onto Endpoints

We now look at what happens at each endpoint type in relation to the security system on each target machine when the security profile is distributed. See “Examples of TME Security Records Mapped onto TACF” on page 117 for much more detail on the UNIX/TACF mapping.

The endpoint is responsible for translating the information in the security profile records into the system-level security entries. The following sections show how the various profile records are translated.

5.13.2.1 Adding Security Group Records

This translation of security group records depends on the type of operating system as follows:

- UNIX/TACF

  It creates a TACF group if it does not already exist. The TACF group name will be the same as the security group record’s specific UNIX name if it is different from the record’s global name.

Note: A UNIX group with the same name may or may not exist on the endpoint systems. Distribution of the security group records will not affect the UNIX level group entries; UNIX-level groups are not created or changed.

It also adds the member list of the security group as a Userlist to the TACF Group and creates TACF user records for each user. It then adds login time restrictions and audit mode values to each TACF user record.

- NT

  The following steps are performed:

  - For each security group, creates an NT Global Group
  - For each direct or indirect (through the associated TME user account) NT user account member of the security group, assigns it as a member of the NT Global Group
  - For each direct or indirect NT user account member of the security group, applies the login time restrictions settings
  - For each security role which the security group is assigned, adds the NT Global Group to the security role's equivalent NT Local Group membership

Note that as a security group maps to an NT Global group, it only makes sense to manage security groups for NT Server systems, not workstations.

This is where the security group attributes map to the system-level security attributes. The TME 10 attributes are used only by the TME 10 Security Management product and are not distributed to the endpoints:

Table 7. Security Group Attributes Mapped to Endpoints

Attributes	Global	TME 10	TACF/ UNIX	NT	TME Admin
Description			✓	✓	
Full name	✓		✓		
Group name	✓		✓	✓	
<endpoint type> group name			✓	✓	
Login audit	✓		✓		
Login time restrictions	✓		✓	✓	
Resource access audit	✓		✓	✓	
TME role assignment list		✓			
TME user list	✓	✓	✓	✓	✓
<endpoint type> TME user list	✓		✓	✓	
<endpoint type> user list	✓		✓	✓	

5.13.2.2 Adding Security Resource Records

This translation of security resource records depends on the type of operating system as follows:

- UNIX/TACF

  Creates a TACF resource, if it does not already exist. See "TACF" on page 105 for a description of the TACF classes that map to the security resource type.

Sets the TACF resource’s default access and audit values as defined in the security resource record.

- NT

  Creates an NT resource for each security resource. Then it applies the audit control policies of the security resource to the NT resource. Note that there is no NT resource creation. Unlike TACF, where if a resource did not exist its future creation would be protected, in NT, resources being managed must already exist to be protected.

This is where the security resource attributes map to the system-level security attributes. The TME 10 attributes are used only by the TME 10 Security Management product and are not distributed to the endpoints:

Table 8. Security Resource Attributes Mapped to Endpoints

Attributes	Global	TME 10	TACF/ UNIX	NT
Access time restrictions	✓		✓	
Audit control	✓		✓	✓
Default access	✓		✓	
Description	✓		✓	
Endpoint type		✓		
Resource name		✓		
Resource type			✓	✓
Role list		✓		
TCP service access	✓		✓	
Host/network name	✓		✓	
Access permissions	✓		✓	
Warning	✓		✓	

5.13.2.3 Adding Security Role Records

This translation of security role records depends on the type of operating system as follows:

- TACF/UNIX

  Security roles cannot be applied directly to TACF records. The access control lists of the resources that are assigned to a role are updated with the group names of all the security groups who are assigned the role, and the permissions allocated to the group correspond with the security role’s access permissions to the resource. This can perhaps be more easily understood by looking at the TACF entries that are created when a role is distributed. See “Examples of TME Security Records Mapped onto TACF” on page 117 and in particular “Example 3: Create a SecSurity Role” on page 122.

- NT

  The following steps are performed:

  - An NT Local group for each security role is created.

- For each security group in the security role’s list of assigned groups, assign the associated NT Global Group to the NT Local Group.
- For each security resource in the security group, add it either as an NT Right or NT Resource/Permission to the associated NT Local Group.

Note that as a security role is mapped to an NT local group, it is valid for implementation on either NT Servers or Workstations.

This is where the security resource attributes map to the system-level security attributes. The TME 10 attributes are used only by the TME 10 Security Management product and are not distributed to the endpoints.

Table 9. Security Role Attributes Mapped to Endpoints

Attributes	Global	TME 10	TACF/ UNIX	NT
Children		✓		
Description		✓		
Parent		✓		
TME role name		✓	✓	✓
TME group list		✓		
<endpoint type>TME group list			✓	✓
<endpoint type> TME resource access			✓	✓
security profile		✓		
Security profile OID		✓		
Resource name			✓	✓
Resource type			✓	✓
Access permissions			✓	✓
Via type			✓	
Via name			✓	
<endpoint type> resource access			✓	✓
Resource name			✓	✓
Resource type			✓	✓
Access permissions			✓	✓
Via type			✓	
Via name			✓	
endpoint type> resource type access			✓	
Resource type			✓	
Access permissions			✓	
Via type			✓	
Via name			✓	

5.13.2.4 Adding System Policy Records

This translation of security policy records depends on the type of operating system as follows:

- TACF/UNIX

  It updates the default access values of the TACF _default records with the security system policy resource type access values.

  It also sets the login, audit and password policies in TACF from the security system policy record. The values can be displayed in TACF with the command:

  ```
  setoptions list
  ```

- NT

  Updates the NT Security Account Manager (SAM) database to reflect login, audit, password policies, and so on. You can check these in the NT User Manager from the **Policies** menu.

This is where the TME System Policy attributes map to the system-level security attributes. The TME 10 attributes are used only by the TME 10 Security Management product and are not distributed to the endpoints.

Table 10. TME System Policy Attributes Mapped to Endpoints

Attributes	Global	TME 10	TACF/ UNIX	NT
Admin Change Password	✓			✓
Check Password Rules	✓		✓	✓
Description		✓		
Lockout After	✓		✓	✓
Lockout Duration	✓		✓	✓
Lockout Harmless Count	✓		✓	✓
Lockout State	✓		✓	✓
Login Audit	✓			✓
Maximum Concurrent Logins	✓		✓	
Maximum Grace Logins	✓		✓	
Maximum Inactive Days	✓		✓	
Maximum Password Age	✓		✓	✓
Maximum Password Reps	✓		✓	
Minimum Password Age	✓		✓	✓
Minimum Password Alpha	✓		✓	
Minimum Password Alphanums	✓		✓	
Minimum Password Length	✓		✓	✓
Minimum Password Lowercase	✓		✓	
Minimum Password Numeric	✓		✓	
Minimum Password Specials	✓		✓	
Minimum Password Uppercase	✓		✓	

Attributes	Global	TME 10	TACF/ UNIX	NT
Password Expiry Warning	✓			
Password History	✓		✓	✓
Resource Type Access Audit			✓	
Resource Type			✓	
Resource Access			✓	
Resource Type Access rights			✓	
Resource Type			✓	
Default Access			✓	
State			✓	
System Policy Name		✓		

5.14 Integration with Standard TME Framework Features

TME 10 Security Management provides the following functions that integrate into the standard TME 10 Framework:

- Authorization roles
- Security tasks
- Notice groups

Each of these will be discussed in the following sections.

5.14.1 Authorization Roles

TME 10 Security Management creates new authorization roles for TMRs and policy regions:

- security_admin

  This assigns authority to administer access to system-level resources. Here are some of the functions that an administrator with this role can perform:

  – Create and modify security group, security role and security resource records
  – Create and modify System Policy records
  – Distribute security profiles

- security_auditor

  This assigns authority to audit the use of system resources and to control which security events are logged. This role is required to execute the Audit Report Generator Task documented in "Security Tasks" on page 103.

- security_operator

  This assigns authority to view all security resource data.

Existing authorization roles still apply to some functions related to managing security profiles. For example, assigning security profiles as managed resources in a policy region requires senior authorization role in the policy region.

Full details of the authorization roles required to perform each function are documented in the *TME 10 Security Management User's Guide*.

5.14.2 Security Tasks

TME 10 Security Management provides a set of tasks that allow administrators to run security jobs. These tasks and jobs behave in the same way as all other tasks and jobs in allowing the execution characteristics to be changed, for example:

- On which systems the tasks/jobs will run
- Where the output is displayed
- Whether to run serially, in parallel or staged in groups

The following tasks are provided:

- TME 10 Security Audit Report Tasks
    - Endpoint Audit Report
    - TME Security Database Report
- TACF Tasks
- Add/Remove TACF Auditor/Administrator
- Backup TACF Database
- Disable Concurrent Logins
- Enable/Disable TACF Trace
- Extract TACF Trusted Programs
- Restore TACF Database
- Show TACF Auditors/Administrators
- Start TACF Servers
- Stop TACF Servers
- Synchronize TACF/UNIX Users & Groups
- TACF root Compliance Report
- TACF Database Maintenance
- TACF Runtime Statistics Report
- TACF/UNIX Integrity Report
- Terminate TACF User

These tasks are all well-described in the *TME 10 Security Management User's Guide*.

5.14.3 Notice Groups

TME 10 Security Management creates a new Notice Group. The following operations are posted in the Security Notice Group:

- TME 10 security Management install
- All operations on a security profile and its records

Each notice will include:

- A unique ID
- Severity level

- Date and time stamp
- TME administrator’s name
- Details of the operation

5.15 Integration with Other TME 10 Products

TME 10 Security Management is integrated with several TME 10 products that you may already be familiar with:

- TME 10 User Administration
- TME 10 Distributed Monitoring
- TME 10 Enterprise Console

These are discussed in the following sections.

5.15.1 TME 10 User Administration

Many customers who install TME 10 Security Management already have TME 10 User Administration installed to manage their user IDs and groups on UNIX, Windows NT and NetWare. However, TME 10 User Administration is not a prerequisite for TME 10 Security Management. The two products can exist independently.

Where TME 10 User Administration is not installed, users to be added as members of a security group can be selected from a UNIX User List, from an NT User List and once supported, from a RACF User List.

Where TME 10 User Administration is installed, users already defined in user profiles are also available as potential members of security groups. On the panel displayed to add users to a security group, the TME User List shows you a list of user profiles that are available and by selecting one of those, you will see the users in that user profile.

Where users are added to user profiles in this way, distribution of the user profile must occur before distribution of the security profile.

5.15.2 Integration with TME 10 Distributed Monitoring

TME 10 Security Management provides monitors for use in TME 10 Distributed Monitoring as follows:

- Availability of TACF server
- Availability of TACF Log Routing server
- Size in bytes of security service audit file
- Size in bytes of security service audit log file
- Size in bytes of free disk space in security audit directory

As with all monitors, you are able to specify actions for different levels of severity. These actions can include sending events to the TME 10 Event Server.

See Figure 45 on page 143 regarding installation of TACF monitors.

5.15.3 Integration with TME 10 Enterprise Console

TME 10 Security Management can send events to the TME 10 Enterprise Console (TEC). A full list of the events supported can be found in the *TME 10 Security Management User Guide*, but here is a summary.

Audit log events are monitored by the TME 10 Enterprise Console Logfile Adapter to format and send events to the TEC. The events selected from the audit logs include:

- Login successes and failures
- File access successes and failures
- Untrust critical file - where a file that has been registered as a trusted file has been flagged as changed and can no longer be trusted.
- Untrust program - where a program that has been registered as a trusted program has been flagged as changed and can no longer be trusted
- Incoming/outgoing network connection successes and failures
- Substitution to system user ID (UNIX root) successes and failures
- Substitution to a non-system user IDs successes and failures

Security Monitor Events:

- Events are generated from the monitors described in "Integration with TME 10 Distributed Monitoring" on page 104.

Security Task Events:

- Events are generated from the tasks described in "Security Tasks" on page 103.

Security Correlation Rules:

- TME 10 Security management supplies correlation rules to the TME 10 Enterprise Console to determine if there are relationships between separate events and what actions to take, for example:

  Any failed substitution to root, from the same user, on any host where they occur three or more times in 30 minutes, triggers the event's severity being raised to CRITICAL.

See Figure 45 on page 143 regarding installation of TACF event integration.

5.16 TACF

TACF stands for Tivoli Access Control Facility. It provides a low-overhead layer on UNIX systems that intercepts system calls. It is based on the product SeOS Access Control Version 2 from Memco Software, Inc. TACF is fully integrated into the TME 10 Security Management architecture and allows you to treat many flavors of UNIX in the same way as Windows NT, and ultimately RACF.

The distribution of TME security profiles customizes the TACF layer with group, resource and access information. This provides TACF with the relevant information required when it checks whether access to resources should be allowed or denied.

TACF intercepts all security-related events, including:

- File open
- File execute
- File delete
- Change of permission
- Change of owner
- Change of group

TACF then checks the entries in the TACF database to establish the access to a resource that the requesting user is allowed. Note that TACF does not replace normal UNIX access control; it sits above it.

For example, TACF may determine that a user has Write access to a file, but the UNIX file permissions may give that user only Read access. So the user will not be able to write to the file. If TACF determines that a user has Read access to a file, where UNIX permissions allow the user to Read and Write, the TACF levels of access prevail and the user is only allowed to Read.

TACF sets the maximum level of access that a user has to a resource, but still needs UNIX to permit the same level of access.

5.16.1 TACF Installation

TACF is supplied as a separate component from the TME 10 Security Management code. It is installed with the standard TME 10 Framework install product facilities.

The installation process creates a UNIX user ID for tmesec, if it does not already exist, and it defines certain key resources and records in the TACF database:

- TACF user records for tmesec, root and nobody
- TACF group entries for:
  - _abspath - Members in group are not allowed to have a dot in their PATH.
  - _restricted - Any FILE access of members in the group is checked.

  These group entries are not directly accessible through TME 10 Security Management, but can be manipulated through TACF commands.
- Default records in classes FILE, CONNECT, TCP, PROCESS, TERMINAL and SURROGATE

  The default access values in the _default records are the values that reflect the resource type access policy of the security system policy records. These values can therefore be changed by distributing security system policy records with different default access for the resource types.
- TACF entry in class SURROGATE for USER.tmesec. This prevents any user from accessing tmesec from an su command.

```
Data for SURROGATE 'USER.tmesec'
-----------------------------------------------------------
Defaccess         : None
Audit mode        : Failure
Owner             : tmesec
Create time       : 03-Jun-1997 11:53
Update time       : 03-Jun-1997 11:53
Updated by        : root

Data for SURROGATE '_default'
-----------------------------------------------------------
Defaccess         : R
Audit mode        : None
Owner             : tmesec
Update time       : 04-Jun-1997 10:15
Updated by        : tmesec
```

- TACF file records for some key UNIX files, such as /etc/passwd and /etc/security/passwd.

```
Data for FILE '/etc/passwd'
-----------------------------------------------------------
Defaccess         : R, W, X, Cre, Del, Chown, Chmod, Utime, Sec, Rename
Audit mode        : Failure
Notify            : root
Owner             : tmesec
Create time       : 03-Jun-1997 11:53
Update time       : 03-Jun-1997 11:53
Updated by        : root

Data for FILE '/etc/security/passwd'
-----------------------------------------------------------
Defaccess         : R, W, X, Del
Warning           : No
Audit mode        : Failure
Notify            : root
Daytime           : Anyday 00:00 to 23:59
Owner             : tmesec
Create time       : 03-Jun-1997 11:53
Update time       : 04-Jun-1997 10:14
Updated by        : tmesec
```

- TACF entry in class TERMINAL for the system on which TACF is installed

  This entry sets the default access for the terminal to Read, and it adds tmesec and root to the access control list with Read and Write. The Read and Write access is required to give these users administrative authority in TACF.

  Users who have only Read access will receive this message if they try to start up the command line interface for TACF:

```
ERROR: Initialization failed, EXITING!!!
(localhost)
ERROR: Login procedure failed
ERROR: You are not allowed to administer this site from terminal
rl0230b.itsc.austin.ibm.com
```

```
Data for TERMINAL 'rl0230b.itsc.austin.ibm.com'
-----------------------------------------------------------
Defaccess         : Read
Warning           : No
Acls              :
    Accessor                Access
    root          (USER   ) R, W
    tmesec        (USER   ) R, W
Audit mode        : Failure
Daytime           : Anyday 00:00 to 23:59
Owner             : tmesec
Create time       : 03-Jun-1997 11:53
Update time       : 04-Jun-1997 11:19
Updated by        : tmesec
```

5.16.2 TACF Database and Classes

The TACF database has two types of records:

- Accessors, for example users and groups
- Resources, for example files and services

Each record defined to TACF belongs to a *class.* There are many classes supplied with TACF, but not all of them are currently used in the TME 10 Security Management environment. Each class has a number of fields, and again not all the fields are currently utilized by TME 10 Security Management. In some areas, you may be able to do your own thing through the command line, but you would have to careful to watch future releases and updates where some of the additional classes and fields could be utilized.

In general, resource records have an *Access Control List (ACL)* which defines the accessors (users and groups) who are allowed to access that object. An exception to this is the TCP class, which we discuss later. Resource records may also contain a default access field. This value is checked when a user or group does not appear in the resource's access control list.

In the following section, we show you the main TACF classes that are used, the fields that are particularly relevant and their relationship to TME 10 Security Management. All the classes are described in detail in the *TME 10 Security Management TACF Reference Guide.*

The classes we look at in particular are:

- GROUP
- USER
- CONNECT
- FILE
- PROCESS
- PROGRAM
- SURROGATE
- TCP
- Gtme_res
- tme_res

Every record in all the classes contains some common fields, for example the owner of the record, the date and time it was created, the date and time it was updated and by whom, and the date and time it was last accessed and by whom.

GROUP and USER classes define accessors rather than resources to be accessed. Groups and Users do not have ACLs; they appear in the ACLs of other classes. See the example of creating a security group in "Examples of TME Security Records Mapped onto TACF" on page 117.

5.16.2.1 GROUP

This is an accessor class rather than a resource class. It does not have an access control list, but it appears in ACLs of resource classes such as FILE.

The GROUP class represents groups of users. Each record includes:

- Name of the group
- List of users belonging to the group

TACF groups are directly related to security groups. When a new security group is distributed to subscribers, a new TACF group is created if it does not exist and modified with the latest list of member users if it does exist.

5.16.2.2 USER

This is an accessor class rather than a resource class. It does not have an access control list, but it appears in ACLs of resource classes such as FILE.

Each user is represented by a record in this class. The record includes:

- User name
- Day and time restrictions
- Audit mode
- The groups to which it is connected

TACF user records are created and modified by the distribution of security group records. A record is also created in the USER class for the group name prefixed with *tme_*.

For example, a group called *Redbook_Authors* has an equivalent USER entry of *tme_Redbook_Authors*. This record does not have any group connections, but it will include the day and time restrictions and audit mode values that apply to all the members of the group.

5.16.2.3 CONNECT

This provides protection for outgoing connections such as Telnet from the local machine. The name of the record is the name of the system being connected to, for example, hostname rl0230a.itsc.austin.ibm.com. Each record includes:

- Day and time restrictions
- Audit mode
- List of accessors allowed to access host and permitted access level
- The default access

TACF CONNECT classes directly relate to security resources with resource type CONNECT.

See the example of creating a security resource with Resource type CONNECT in “Examples of TME Security Records Mapped onto TACF” on page 117.

5.16.2.4 FILE

Every object in the FILE class defines the access allowed to a file or directory. Each record includes:

- Day and time restrictions
- Audit mode
- List of accessors allowed to access the file and their permitted access
- Default access
- Conditional ACL when accessed through a program defined in PROGRAM class

TACF FILE directly relates to security resources with resource type FILE. See the example of creating a security resource with Resource type FILE in “Examples of TME Security Records Mapped onto TACF” on page 117.

5.16.2.5 PROCESS

This represents programs that need to be protected from being killed. The program (the file) must be defined in the FILE class before it can be defined here. Each record includes:

- Day and time restrictions
- Audit mode
- List of accessors allowed to kill process
- Default access

TACF PROCESS directly relates to security resources with resource type PROCESS.

5.16.2.6 PROGRAM

This class defines programs that can be trusted not to have security breaches. Each record includes:

- Day and time restrictions
- Audit mode
- List of accessors and their permissions
- Default access

TACF PROGRAM directly relates to security resources with resource type PROGRAM for resources marked as setuid or setgid in the UNIX file system. This class is also updated automatically with a program name when conditional access to resources through a program is specified in security role records.

5.16.2.7 SECFILE

This class is similar to the PROGRAM class except that SECFILE objects do not appear in a conditional access control list.

TACF SECFILE directly relates to security resources with the resource type SECFILE.

5.16.2.8 SURROGATE

This defines restrictions that protect users from an su request by another user. Each record includes:

- Day and time restrictions
- Audit mode
- List of accessors allowed to su to a user
- Default access
- Conditional ACL when accessed through a program defined in PROGRAM class

TACF SURROGATE directly relates to security resources with resource type SURROGATE.

5.16.2.9 TERMINAL

This defines terminals on the local host. The TERMINAL class is checked to see if the user is authorized to login from that terminal. Each record includes:

- The name of the terminal, for example rl0230b.itsc.austin.ibm.com
- Day and time restrictions
- Audit mode
- List of accessors allowed to login from terminal
- Default access
- Conditional ACL when accessed through a program defined in PROGRAM class

TACF TERMINAL directly relates to security resources with resource type TERMINAL.

5.16.2.10 TCP

This class governs the access that hosts, hosts on a particular network and hosts with similar names, have to services on the local host. The services that can be protected are those defined in /etc/services. Each record includes:

- Name of service
- List of hosts that can access the service
- Audit mode
- Default access
- Day and time restrictions
- Audit mode

TACF TCP class directly relates to security resources with resource type TCP.

5.16.2.11 Gtme_res

There is a record in this class for each security role defined by TME 10 Security Management. The member list of each Gtme_res represents all the resources, along with the access rights to those resources, to which the role has access. The access control list contains a list of groups that have this role assigned.

Each record includes:

- Name of the role
- List of groups that have role assigned
- Member list of resources assigned to the role

TACF Gtme_class directly relates to security roles.

5.16.2.12 tme_res

Each record represents a resource that is defined in TACF, combined with a unique access level built into its name. In addition, there is any conditional access level through a PROGRAM. These resources are defined as members of Gtme_res resources (roles).

The tme_res records also contain a list of Gtme_res (roles) resources to which they are assigned. This list appears under the heading “Groups”. This is a little difficult to explain, but the examples we show later in “Examples of TME Security Records Mapped onto TACF” on page 117 show how tme_res, Gtme_res and security roles fit together.

Each record includes:

- Name, for example CONNECT:_default:Read:PROGRAM: or FILE:/tme/redbook/*:Read.Write:PROGRAM:
- Roles in which record is a member

5.16.3 TACF Commands

Under normal circumstances, it is not necessary to interface directly with TACF from the command line. Instead, TME 10 Security Management adds a number of commands to the TME environment such as `wcrtsec`. These are what you would use for normal command-line activities. However, you might find it useful in problem determination in circumstances where you are perhaps getting unexpected access or denial of access to resources.

The user ID tmesec owns the TACF database and most of the associated files. It is defined as a TACF user with attributes that allow it to add, modify and delete resources and accessors. Other user IDs, for example root, may have the authority to start-up the command line environment, but be restricted to viewing resources and accessors. You can specify additional users with full authority to administer TACF at install time under the “Install Options” (see Figure 38). Alternatively, for users that are already defined in TCAF, you can enter the following from the TACF command line:

```
authorize terminal <TACFhostname> uid(<username>) access(read,write)
```

```
chusr <username> admin
```

The first command allows the specified user ID authority to start the command line interface from the server. This is described in Section 5.16.1, “TACF Installation” on page 106, and the section on the default entry made in the TERMINAL class. The second command gives the specified user administrative authority. You can see the effect of this command by entering the following command from the TACF CLI:

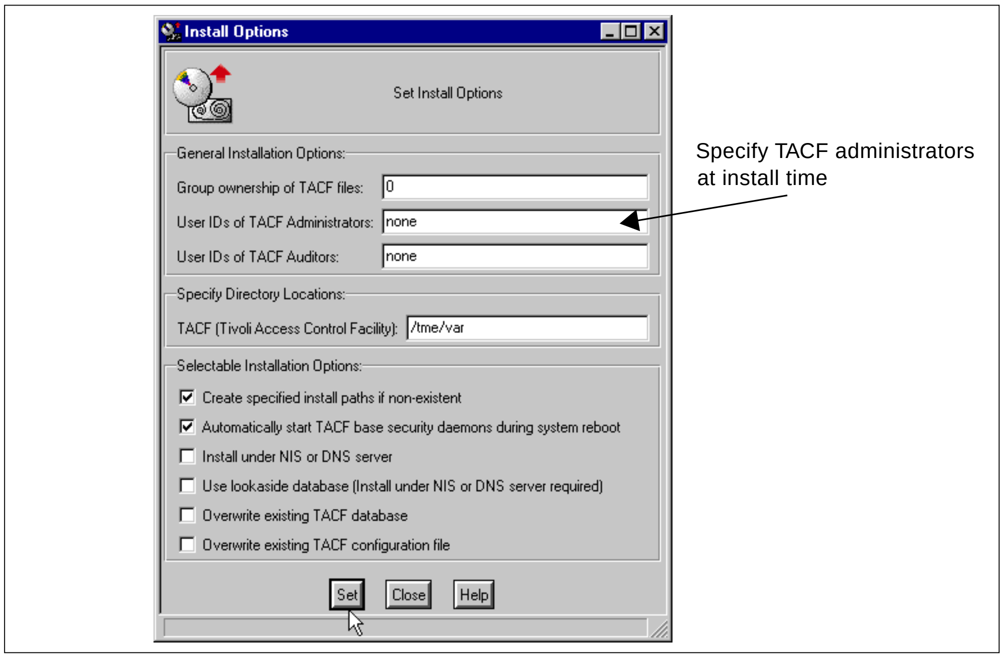
```
su <username>
```

You should see the following entry in the user’s data:

```
User Mode              : Admin
```

To remove administrative authority, enter:

```
chusr <username> admin-
```


Install Options

Set Install Options

General Installation Options:

Group ownership of TACF files: 0

User IDs of TACF Administrators: none

User IDs of TACF Auditors: none

Specify Directory Locations:

TACF (Tivoli Access Control Facility): /tme/var

Selectable Installation Options:

- [✓] Create specified install paths if non-existent
- [✓] Automatically start TACF base security daemons during system reboot
- [ ] Install under NIS or DNS server
- [ ] Use lookaside database (Install under NIS or DNS server required)
- [ ] Overwrite existing TACF database
- [ ] Overwrite existing TACF configuration file

Figure 38. TACF Install Options Panel - Define TACF Administrators

TME 10 Security Management also provides tasks to add and remove TACF administrators and to terminate a TACF user. See the *TME 10 Security Management User's Guide* for further information on these tasks.

TACF commands are available to run in two environments:

- TACF command shell

  These commands interface directly with the objects and resources defined in the TACF database.

- UNIX environment

  These commands interface with the UNIX operating system, allowing you to add, modify, and delete UNIX users and groups and modify file permissions.

To access the TACF command line:

1. Log in as *tmesec*
2. Set up the TME 10 environment:

   ```
   . /etc/Tivoli/setup_env.sh
   ```

3. Enter: `selang`

   This gives you the TACF prompt:

   ```
   TACF>
   ```

4. If the TACF daemons are not running, you can enter:

   ```
   sedlang
   ```

5. To change to the UNIX environment, enter:

   ```
   environment unix
   ```

   This changes the prompt to:

   ```
   TACF(unix)>
   ```

6. To get a list of all commands available in the UNIX environment, enter:

   `TACF(unix)> help`

```
        Command listing for the unix environment:
        chfile                 - change file attributes (mode,suid/sgid bits etc)
        chgrp                  - change an existing group in the unix database.
        chusr                  - change user attributes in unix database.
        editgrp    (eg)        - if group nonexistent operates as newgrp
                                 if group exists      operates as  chgrp
        editusr    (eu)        - if user  nonexistent operates as newusr
                                 if user  exists      operates as  chusr
        environment            - change the target environment.
        history                - show command history information
        join                   - join user to group.
        join-                  - remove user from group.
        newgrp                 - add a new group in unix environment.
        newusr                 - add a new user to the unix database.
        rmgrp                  - remove a group from the unix environment.
        rmusr                  - remove a user from unix environment.
        showfile               - list file attributes.
        showgrp                - list group's data.
        showusr                - list user's data.
```

7. To return to the TACF environment, enter:

   `environment seos`

8. To get a list of all commands available in the TACF environment, enter:

   `TACF> help`

```
        Command listing :
        ---------------
        alias                  - define/display a pseudonym.
        authorize  (auth)      - set user/group's permissions to a resource.
        authorize- (auth-)     - remove user/group's permissions to a resource.
        chfile     (cf)        - change a file profile in the TACF database
        chgrp      (cg)        - change group attributes.
        chres      (cr)        - change resource attributes.
        chusr      (cu)        - change user attributes.
        editfile   (ef)        - if file     nonexistent operates as newfile
                                 if file     exists      operates as  chfile
        editgrp    (eg)        - if group    nonexistent operates as newgrp
                                 if group    exists      operates as  chgrp
        editres    (er)        - if resource nonexistent operates as newres
                                 if resource exists      operates as  chres
        editusr    (eu)        - if user     nonexistent operates as newusr
                                 if user     exists      operates as  chusr
        environment            - change the target environment.
        find       (f)         - show a listing of profiles in a class.
        help [topic]           - show help [concerning topic].
        history                - show command history information.
        hosts [(hosts-list)]   - show/set list of target hosts/PMDBs.
        join       (j)         - join a user to a group.
        join-      (j-)        - remove a user from a group.
        newfile    (nf)        - add a file profile to the TACF database
        newgrp     (ng)        - add a new group to the TACF database.
        newres     (nr)        - add a new resource to TACF database.
        newusr     (nu)        - add a new user to the TACF database.
        rmfile     (rf)        - remove a file profile from the TACF database
        rmgrp      (rg)        - remove a group from the TACF database.
        rmres      (rr)        - remove a resource from TACF database.
        rmusr      (ru)        - remove a user from the TACF database.
        ruler                  - select properties to display in query.
        setoptions (so)        - set/show TACF database options.
        showfile   (sf)        - list a file from the TACF database.
        showgrp    (sg)        - list a group from the TACF database.
        showres    (sr)        - list a resource.
        showusr    (su)        - list a user from the TACF database.
        source                 - read commands from a file.
        unalias                - remove a pseudonym defined by alias.
```

In the next section, we look at a few useful commands, although mainly you will interface with TACF through the TME 10 Security Management GUI. You can get help on any specific command by using `help commandname`.

5.16.3.1 UNIX Environment Commands

The `showusr (su) <username>` command creates the following output:

```
TACF(unix)> showusr lynn
(localhost)
Unix :
======
Data for USER 'lynn'
-----------------------------------------------------------
User id           : 202
Primary group     : staff
Group id          : 1
Shell program     : /usr/bin/ksh
Home directory    : /home/lynn
Gecos             : lynn
```

The `showfile (sf) <filename>` command creates the following output (Note: wildcards(*) are not supported in this command):

```
TACF(unix)> sf /usr/local/Tivoli
(localhost)
Unix :
======
Data for FILE '/usr/local/Tivoli'
-----------------------------------------------------------
User access       : rwx
Group access      : r-x
Other access      : r-x
Owner             : sys
Group id          : sys
Setuid            : No
Setgid            : Yes
Size              : 512
Device            : 655372
Inode             : 2
Directory         : Yes
Link name         :
Last access time  : 21-May-1997 15:58
Last inode change : 05-May-1997 09:08
Last modified     : 05-May-1997 09:08
```

5.16.3.2 TACF Environment Commands

We look at a few useful commands here and later in "Examples of TME Security Records Mapped onto TACF" on page 117, we include the TACF commands that are generated when the TME security records are distributed.

- `find (f)` – Displays all TACF classes (not shown)
- `showres (sr) <classname>` – Displays the field names in a class

  ```
  TACF> showres CONNECT
  (localhost)
  Ruler for class CONNECT contains:
  -----------------------------------
  UACC
  WARNING
  ACL
  PACL
  RAUDIT
  NOTIFY
  DAYTIME
  GROUPS
  OWNER
  SECLEVEL
  SECLABEL
  CATEGORY
  CREATE_TIME
  UPDATE_TIME
  UPDATE_WHO
  COMMENT
  ```

- `showres (sr) <classname> <resourcename>` – The *resourcename* can include wildcards(*) or multiple names in parentheses.

```
TACF> showres CONNECT rl*
(localhost)
Data for CONNECT 'rl0230a.itsc.austin.ibm.com'
-----------------------------------------------------------
Defaccess         : None
Acls              :
    Accessor               Access
    Redbook_Authors(GROUP  ) R

Audit mode        : Failure
Owner             : tmesec
Create time       : 16-May-1997 11:57
Update time       : 21-May-1997 11:42
Updated by        : tmesec
```

- `showfile` <resourcename> —
- `showgrp` <groupname> —
- `showusr` <username> —

```
TACF> showusr root
(localhost)
Data for USER 'root'
-----------------------------------------------------------
User mode         : Auditor
Audit mode        : Failure, Login-Failure
Daytime           : Anyday 00:00 to 23:59
Group connections :
    Group         Owner       Date        Time  Type
    tme_sec       root        05-May-1997 09:21 Regular
Owner             : tmesec
Maxlogins         : 0
Last accessed     : _CRONJOB_
Last access time  : 21-May-1997 12:00
Update time       : 05-May-1997 14:09
Updated by        : tmesec
```

- `setoptions (so) list` – This lists the systemwide TACF options. Many of the values relate directly to the attributes defined in the security system policy records. For example:

  TCP   : Yes

  shows that the resource type TCP is set to *Enabled* in the system policy record. This means that checking is done against TCP resources.

```
TACF> setoptions list
(localhost)
Data for SeOS options
-----------------------------------------------------------
Password rules    :
        Alpha             : 1
        Alpha numeric     : 1
        Gracelogins       : 6
        History           : 0
        Interval          : 40
        Password min life : 0
        Length            : 5
        Lower             : 1
        Max rep           : 2
        Numeric           : 1
        Special           : 1
        Upper             : 1
        Old PW check      : Yes
        Name check        : Yes
Inactive days     : 0
Admin             : Yes
Appl              : Yes
Auth-host         : Yes
Category          : Yes
Connect           : Yes
File              : Yes
Host              : No
Password          : Yes
Process           : Yes
Program           : Yes
```

```
Password policy   : Yes
Seclabel          : No
Seclevel          : Yes
Sudo              : Yes
Surrogate         : Yes
TCP               : Yes
Terminal          : Yes
UDP               : Yes
tme_res           : No
Last startup      : 21-May-1997 09:38
Last shutdown     : 21-May-1997 09:34
Update time       : 13-May-1997 11:05
Updated by        : tmesec
```

- `setoptions class-(TCP)` - This disables checking against the TCP resource type.

5.16.4 Examples of TME Security Records Mapped onto TACF

We looked in some detail in "How Access is Granted/Denied" on page 88 at how decisions are made to deny or allow access to resources. The following table shows how the TME 10 Security Management access levels are mapped onto TACF:

Table 11. Mapping of TME 10 Security Management Access to TACF

TME 10 Security Management Access	TACF Access
Default access defined for each TME 10 Security Management resource	Defaccess field in TACF resource records
Access defined for a resource when assigned to a role	ACL field in TACF resources shows the group names assigned to the role with access level
Default resource type access defined for a role	ACL field in _default record for each class, such as CONNECT, shows the group names assigned to the role with access level
Default resource type access defined in system policy record	Defaccess field in _default record for each class

Here are some examples of the TACF entry resulting from the mapping of TME 10 Security Management access levels to TACF (produced with the `showres/sr` command).

- Default access in a security resource record maps to the *defaccess* field in the TACF resource record:

```
Data for FILE '/tmp/testfile1'
-----------------------------------------------------------
Defaccess         : None
Audit mode        : Failure
Daytime           : Anyday 00:00 to 23:59
Owner             : tmesec
Create time       : 03-Jun-1997 12:13
Update time       : 03-Jun-1997 12:13
Updated by        : tmesec
```

- Access permissions in security role records for specific resources map to the access control lists of TACF resources, where the list of groups in the TACF ACL matches the security groups that have the role assigned:

```
Data for FILE '/tme/redbook/*'
-----------------------------------------------------------
Defaccess         : None
Acls              :
    Accessor                Access
   Redbook_Authors(GROUP  ) R, W
Audit mode        : Failure
Daytime           : Anyday 00:00 to 23:59
Owner             : tmesec
Create time       : 04-Jun-1997 10:14
Update time       : 04-Jun-1997 12:13
Updated by        : tmesec
```

- System policy records for default access to resource types map to the default access level of the _default record in TACF classes that match the security resource type:

```
Data for SURROGATE '_default'
-----------------------------------------------------------
Defaccess         : R
Audit mode        : None
Owner             : tmesec
Update time       : 04-Jun-1997 10:15
Updated by        : tmesec
```

- Security role records for default access to resource types map to the access control lists of the _default record in TACF. The list of groups in the ACL matches the security groups that have the role assigned:

```
Data for PROCESS '_default'
-----------------------------------------------------------
Defaccess         : R
Acls              :
    Accessor                Access
   Redbook_Authors(GROUP  ) None
Audit mode        : None
Owner             : tmesec
Update time       : 04-Jun-1997 10:15
Updated by        : tmesec
```

There now follow some examples of TACF database entries resulting from the distribution of new and modified TME 10 Security Management records. We also show you the TACF commands that are generated at distribution.

5.16.4.1 Example 1: Create a Security Group

Create a security group with the following attributes:

UNIX Group Name	Redbook_Authors
Description	ITSO Residents writing Redbooks
Member List	lynn

The following TACF commands to create the entries are generated by the distribution:

```
newgrp ('Redbook_Authors') owner ('tmesec') comment ('ITSO Residents writing
  Redbooks')
```

```
newusr ('tme_Redbook_Authors') owner ('tmesec') restrictions (days(anyday)
  time(anytime) ) audit (failure loginfail)

newusr ('lynn') owner ('tmesec')

join ('lynn') group ('Redbook_Authors')

chusr ('lynn') owner ('tmesec') restrictions (days(anyday) time(anytime) )
  audit (failure loginfail)
```

After distribution to a TACF/UNIX system, you have these entries in TACF:

```
showgrp Redbook_Authors
(localhost)

Data for GROUP 'Redbook_Authors'
-----------------------------------------------------------
Userlist          :
    lynn
Owner             : tmesec              <-------------  Owner of TACF
Create time       : 20-Sep-1997 17:08                    database
Update time       : 20-Sep-1997 17:08
Updated by        : tmesec
Comment           : ITSO Residents writing Redbooks

showusr ('lynn')
(localhost)

Data for USER 'lynn'
-----------------------------------------------------------
Audit mode        : Failure, Login-Failure  <---------  Default Audit
Daytime           : Anyday 00:00 to 23:59                attributes
Group connections :
    Group         Owner       Date        Time  Type
  Redbook_Authors tmesec      20-Sep-1997 17:08 Regular
Owner             : tmesec
Create time       : 20-Sep-1997 17:08
Update time       : 20-Sep-1997 17:09
Updated by        : tmesec

showusr ('tme_Redbook_Authors')
(localhost)

Data for USER 'tme_Redbook_Authors'
-----------------------------------------------------------
Audit mode        : Failure, Login-Failure            Notice we also get
Daytime           : Anyday 00:00 to 23:59             a user entry of the
Owner             : tmesec                            Group name prefixed
Create time       : 20-Sep-1997 17:08                 with tme_ but the
Update time       : 20-Sep-1997 17:09                 Group connection
Updated by        : tmesec                            does not apply.
```

Now change the security group Redbook_Authors audit attributes:

Audit Control   Success, Failure, Loginsuccess, Loginfail

The following TACF commands to create the entries are generated by the distribution:

```
chusr ('tme_Redbook_Authors') owner ('tmesec') restrictions(days (anyday)
  time(anytime) ) audit( success loginsuccess )
```

```
chusr ('lynn') audit( success loginsuccess )
```

The TACF entry for user lynn now shows:

```
showusr ('lynn')
(localhost)
Data for USER 'lynn'
-----------------------------------------------------------
Audit mode        : Success, Login-Success
Daytime           : Anyday 00:00 to 23:59
Group connections :
    Group         Owner       Date        Time  Type
 Redbook_Authors  tmesec      20-May-1997 17:08 Regular
Owner             : tmesec
Create time       : 20-May-1997 17:08
Update time       : 20-May-1997 17:31
Updated by        : tmesec

showusr ('tme_Redbook_Authors')
(localhost)
Data for USER 'tme_Redbook_Authors'
-----------------------------------------------------------
Audit mode        : Success, Login-Success
Daytime           : Anyday 00:00 to 23:59  <--  Notice that the audit mode
Owner             : tmesec                      also changed in the entry
Create time       : 20-May-1997 17:08           for tme_Redbook_Authors
Update time       : 20-May-1997 117:31
Updated by        : tmesec
```

Now we illustrate that where a user is a member of more than one group, the audit values are cumulative. This is described in Section 5.5.4, "Audit Control" on page 67.

Let's add Lynn as a member of another security group called Customer_support, where there are the following audit attributes:

Audit Control	Failure, Loginfail

The following TACF commands are generated at distribution:

```
newgrp ('Customer_Support') owner ('tmesec')

newusr ('tme_Customer_Support') owner ('tmesec') restrictions(days( anyday )
  time( anytime )) audit( failure loginfail )

join ('lynn') group ('Customer_Support')
```

The TACF entries now show:

```
showgrp ('Customer_Support')
(localhost)
Data for GROUP 'Customer_Support'
--------------------------------------------------
Userlist         :
     lynn
Owner            : tmesec
Create time      : 18-Aug-1997 10:41
Update time      : 18-Aug-1997 10:41
Updated by       : tmesec

showusr ('tme_Customer_Support')
(localhost)
Data for USER 'tme_Customer_Support'
--------------------------------------------------
Audit mode       : Failure, Login-Failure
Daytime          : Anyday 00:00 to 23:59
Owner            : tmesec
Create time      : 18-Aug-1997 10:41
Update time      : 18-Aug-1997 10:41
Updated by       : tmesec

showusr ('tme_Redbook_Authors')
(localhost)
Data for USER 'tme_Redbook_Authors'
--------------------------------------------------
Audit mode       : Success, Login-Success
Daytime          : Anyday 00:00 to 23:59
Owner            : tmesec
Create time      : 15-Aug-1997 15:39
Update time      : 18-Aug-1997 10:24
Updated by       : tmesec
                                        Cumulative effect
showusr ('lynn')
(localhost)
Data for USER 'lynn'
--------------------------------------------------
Audit mode       : Success, Login-Success, Failure, Login-Failure
Daytime          : Anyday 00:00 to 23:59
Group connections :
    Group         Owner       Date        Time  Type
 Customer_Supoort tmesec      18-Aug-1997 10:41 Regular
  Redbook_Authors tmesec      15-Aug-1997 15:39 Regular
Owner            : tmesec
Last accessed    : pegasus
Last access time : 14-Aug-1997 11:53
Create time      : 14-Aug-1997 10:49
Update time      : 18-Aug-1997 10:41
Updated by       : tmesec
```

5.16.4.2 Example 2: Create a Security Resource with Resource Type FILE

Create a security resource as follows:

Endpoint Type	UNIX
Resource Type	FILE
Resource name	tme/redbook/*
Default access	None

Description	Repository for Security Management Redbook

Leave all other attributes as default.

The following TACF command is generated at distribution:

```
nr FILE  ('/tme/redbook/*') comment ('Repository for Security Management
  Redbook') audit( none ) defaccess( None )
  restrictions(days( anyday ) time( anytime )) owner( tmesec )
```

After distribution, you have this entry in TACF:

```
showfile /tme/redbook/*
(localhost)
Data for FILE '/tme/redbook/*'
-----------------------------------------------------------
Defaccess          : None                                  Default value
Audit mode         : Failure   <-------------------        assigned
Daytime            : Anyday 00:00 to 23:59
Owner              : tmesec
Create time        : 20-May-1997 17:42
Update time        : 20-May-1997 17:42
Updated by         : tmesec
Comment            : Repository for Security Management Redbook
```

Change the security resource /tme/redbook/*:

Default Access	Read
Days of week	Mon, Tue, Wed, Thu, Fri
Audit Control	Success, Failure

The following TACF command is generated at distribution:

```
cr FILE  ('/tme/redbook/*') comment ('Repository for Security Management
  Redbook') audit( success failure ) defaccess( Read ) restrictions(days
  ( weekdays ) time( anytime )) warning- owner( tmesec )
```

The TACF entry now shows:

```
sf /tme/redbook/*
(localhost)
Data for FILE '/tme/redbook/*'
-----------------------------------------------------------
Defaccess          : R
Audit mode         : Success, Failure                          Changed
Daytime            : Weekdays 00:00 to 23:59                   values
Owner              : tmesec
Create time        : 20-May-1997 17:42
Update time        : 20-May-1997 17:52
Updated by         : tmesec
Comment            : Repository for Security Management Redbook
```

5.16.4.3 Example 3: Create a SecSurity Role

Create a security role as follows:

Role Name	Write_Redbook
Resource List	/tme/redbook/*

Resource Access Rights	Read, Write
Assign Groups	Redbook_Authors

Leave all other attributes as default.

The following TACF commands are generated at distribution:

```
newres Gtme_res  ('Write_Redbook') owner(tmesec)

authorize Gtme_res Write_Redbook gid('Redbook_Authors')

newres tme_res  ('FILE:/tme/redbook/*:Read.Write:PROGRAM:') owner(tmesec)

chres Gtme_res Write_Redbook mem+('FILE:/tme/redbook/*:Read.Write:PROGRAM:')

authorize  FILE  ('/tme/redbook/*') gid('Redbook_Authors')
  access('Read','Write')
```

The TACF entries now show:

```
showres Gtme_res  ('Write_Redbook')
(localhost)                                                 Resource name is the
Data for Gtme_res 'Write_Redbook'                           same as the role name
-------------------------------------------------
Defaccess         : None
Acls              :
     Accessor                Access                         Role is assigned
      Redbook_Authors(GROUP  ) R                            to this group
Members           :
FILE:/tme/redbook/*:Read.Write:PROGRAM:
Audit mode        : Failure                                    Member list contains
  Owner            : tmesec                                    tme_res resources.
Create time       : 18-Aug-1997 12:18                          The name is made
Update time       : 18-Aug-1997 12:18                          up of -
Updated by        : tmesec                                     type:name:access
                                                               levels:conditional
showres tme_res  ('FILE:/tme/redbook/*:*:PROGRAM:')            access via PROGRAM:
(localhost)
Data for tme_res 'FILE:/tme/redbook/*:Read.Write:PROGRAM:'
-------------------------------------------------
Defaccess         : None
Audit mode        : Failure
Groups            :
      Write_Redbook                                         Role(s) to which this
  Owner             : tmesec                                resource is assigned
Create time       : 18-Aug-1997 12:18
Update time       : 18-Aug-1997 12:18
Updated by        : tmesec

sf /tme/redbook/*
(localhost)
Data for FILE '/tme/redbook/*'
-------------------------------------------------
Defaccess         : R
Warning           : No
Acls              :
     Accessor                Access
     Redbook_Authors(GROUP  ) R, W                          Users in the group have
Audit mode        : Success, Failure                        access to all members
Daytime           : Weekdays 00:00 to 23:59                 listed in roles where
Owner             : tmesec                                  group is in the role's
Create time       : 18-Aug-1997 11:55                       ACL. Access level
Update time       : 18-Aug-1997 12:18                       reflects the levels
Updated by        : tmesec                                  built into the tme_res
Comment           : Repository for Security Management Redbook  resource name.

su lynn
(localhost)
Data for USER 'lynn'
----------------------------------------------------------
Audit mode        : Success, Login-Success, Failure, Login-Failure
Daytime           : Anyday 00:00 to 23:59
Group connections :
   Group         Owner       Date        Time  Type
Redbook_Authors tmesec      15-Aug-1997 15:39 Regular
Owner             : tmesec
Last accessed     : pegasus                                 User has access to all
Last access time  : 14-Aug-1997 11:53                       resources defined in
Create time       : 14-Aug-1997 10:49                       roles assigned to group
Update time       : 18-Aug-1997 12:18
Updated by        : tmesec
```

5.16.4.4 Example 4: Setting the Resource Type Access Levels in a Role

Now let’s suppose that our second group, Customer_Support, that already has user lynn as a member, is now assigned a role called Supporting_Customers. This has Read access to /tme/redbook/*, and we'll assume we've reset the default access for /tme/redbook/* back to None.

TACF commands generated at distribution are:

```
newres Gtme_res  ('Supporting_Customers') owner(tmesec)

authorize Gtme_res Supporting_Customers gid('Customer_Support')

newres tme_res  ('FILE:/tme/redbook/*:Read:PROGRAM:') owner(tmesec)

chres Gtme_res Supporting_Customers mem+('FILE:/tme/redbook/*:Read:PROGRAM:')

authorize  FILE  ('/tme/redbook/*') gid('Customer_Support') access('Read')
```

The TACF entries show:

```
showres Gtme_res  ('Write_Redbook')
(localhost)
Data for Gtme_res 'Write_Redbook'
--------------------------------------------------
Defaccess         : None
Acls              :
      Accessor               Access
      Redbook_Authors(GROUP  ) R

Members           :
      FILE:/tme/redbook/*:Read.Write:PROGRAM:
Audit mode        : Failure
Owner             : tmesec
Create time       : 18-Aug-1997 12:18
Update time       : 18-Aug-1997 12:18
Updated by        : tmesec

showres Gtme_res  ('Supporting_Customers')
(localhost)
Data for Gtme_res 'Supporting_Customers'
--------------------------------------------------
Defaccess         : None
Acls              :
      Accessor               Access
      Customer_Support(GROUP  ) R

Members           :
      FILE:/tme/redbook/*:Read:PROGRAM:
Audit mode        : Failure
Owner             : tmesec
Create time       : 18-Aug-1997 13:38
Update time       : 18-Aug-1997 13:38
Updated by        : tmesec

showres tme_res  ('FILE:/tme/redbook/*:*:PROGRAM:')
(localhost)
Data for tme_res 'FILE:/tme/redbook/*:Read.Write:PROGRAM:'
--------------------------------------------------
Defaccess         : None
Audit mode        : Failure
Groups            :
      Write_Redbook
Owner             : tmesec
Create time       : 18-Aug-1997 12:18
Update time       : 18-Aug-1997 12:18
Updated by        : tmesec

Data for tme_res 'FILE:/tme/redbook/*:Read:PROGRAM:'
--------------------------------------------------
Defaccess         : None
Audit mode        : Failure
Groups            :
      Supporting_Customers
Owner             : tmesec
Create time       : 18-Aug-1997 13:38
Update time       : 18-Aug-1997 13:38
Updated by        : tmesec
```

tme_res resources for the same FILE but different access levels and assigned to different roles

```
(localhost)
Data for FILE '/tme/redbook/*'
-----------------------------------------------------------
Defaccess         : None
Warning           : No
Acls              :
     Accessor               Access
     Customer_Support(GROUP  ) R


     Redbook_Authors(GROUP  ) R, W

Audit mode        : Success, Failure
Daytime           : Weekdays 00:00 to 23:59
Owner             : tmesec
Create time       : 18-Aug-1997 11:55
Update time       : 18-Aug-1997 13:38
Updated by        : tmesec
Comment           : Repository for Security Management Redbook

TACF> su lynn
(localhost)
Data for USER 'lynn'
-----------------------------------------------------------
Audit mode        : Success, Login-Success, Failure, Login-Failure
Daytime           : Anyday 00:00 to 23:59
Group connections :
     Group         Owner       Date        Time  Type
  Customer_Support tmesec      18-Aug-1997 10:41 Regular

   Redbook_Authors tmesec      15-Aug-1997 15:39 Regular
Owner             : tmesec
Last accessed     : pegasus
Last access time  : 14-Aug-1997 11:53
Create time       : 14-Aug-1997 10:49
Update time       : 18-Aug-1997 13:38
Updated by        : tmesec
```

5.16.4.5 Example 5: Setting the Resource Type Access Levels in a Role

In the previous example, we did not specify any values for the Resource type access rights. The default policies of the security role do not set any default values. Where no values are set in TACF, the access decision mechanism simply moves on to the next level of checking.

In this example, we show what happens in TACF when resource type access rights are specified.

Now, set the resource type access rights in security role Write_Redbook as follows:

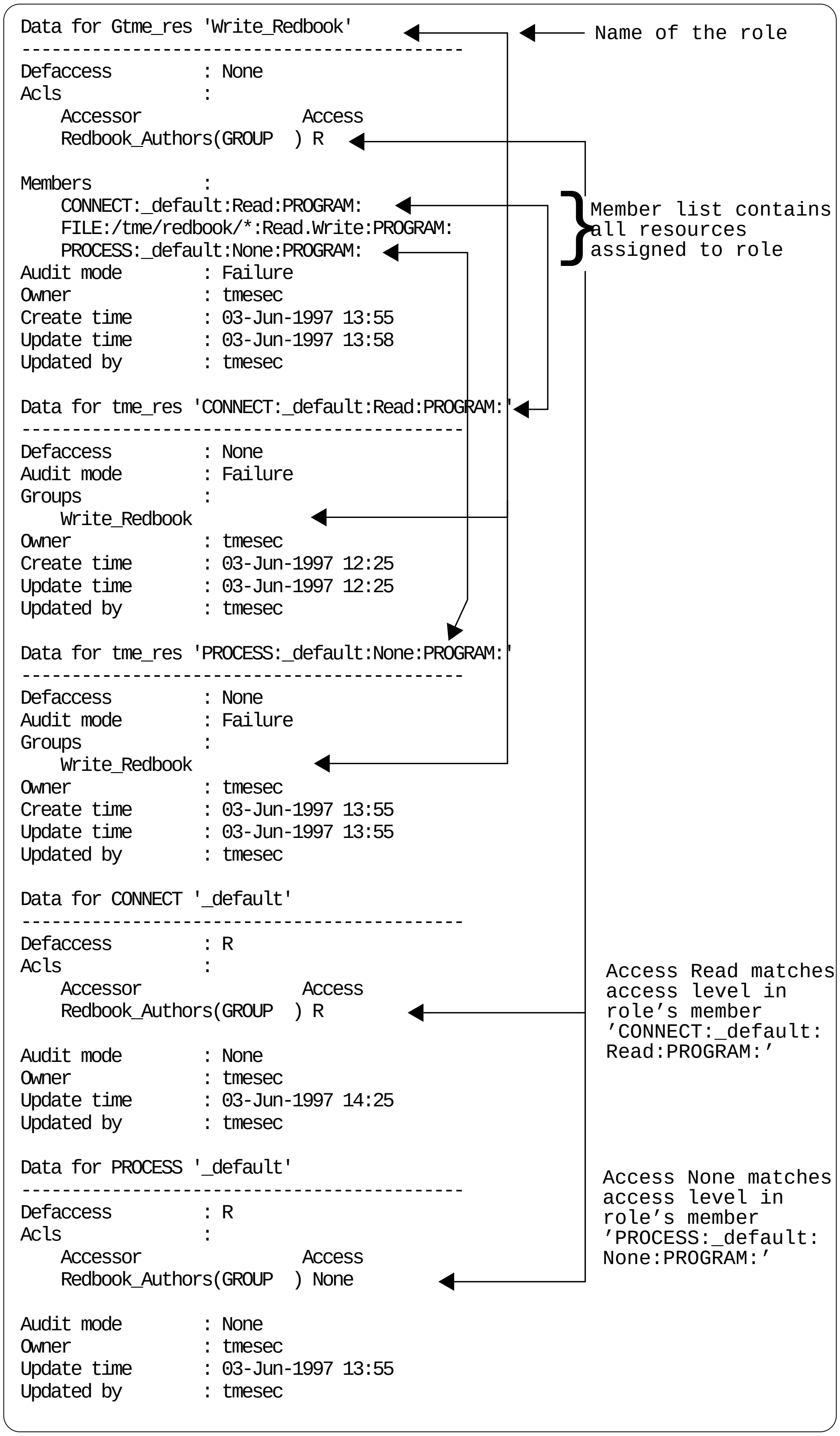
CONNECT	Set to Access
PROCESS	Set to No Access

The following TACF commands are generated by the distribution:

```
newres tme_res  ('CONNECT:_default:Read:PROGRAM:') owner(tmesec)

chres Gtme_res  ('Write_Redbook') mem+('CONNECT:_default:Read:PROGRAM:')
```

```
authorize  CONNECT  ('_default') gid('Redbook_Authors') access('Read')

newres tme_res  ('PROCESS:_default:None:PROGRAM:') owner(tmesec)

chres Gtme_res  ('Write_Redbook') mem+('PROCESS:_default:None:PROGRAM:')

authorize  PROCESS  ('_default') gid('Redbook_Authors') access('None')
```

After distribution, you have these entries in TACF:


Name of the role
Member list contains all resources assigned to role
Access Read matches access level in role's member 'CONNECT:_default: Read:PROGRAM:'
Access None matches access level in role's member 'PROCESS:_default: None:PROGRAM:'

```
Data for Gtme_res 'Write_Redbook'
-------------------------------------------
Defaccess          : None
Acls               :
    Accessor                 Access
    Redbook_Authors(GROUP  ) R

Members            :
    CONNECT:_default:Read:PROGRAM:
    FILE:/tme/redbook/*:Read.Write:PROGRAM:
    PROCESS:_default:None:PROGRAM:
Audit mode         : Failure
Owner              : tmesec
Create time        : 03-Jun-1997 13:55
Update time        : 03-Jun-1997 13:58
Updated by         : tmesec

Data for tme_res 'CONNECT:_default:Read:PROGRAM:'
-------------------------------------------
Defaccess          : None
Audit mode         : Failure
Groups             :
    Write_Redbook
Owner              : tmesec
Create time        : 03-Jun-1997 12:25
Update time        : 03-Jun-1997 12:25
Updated by         : tmesec

Data for tme_res 'PROCESS:_default:None:PROGRAM:'
-------------------------------------------
Defaccess          : None
Audit mode         : Failure
Groups             :
    Write_Redbook
Owner              : tmesec
Create time        : 03-Jun-1997 13:55
Update time        : 03-Jun-1997 13:55
Updated by         : tmesec

Data for CONNECT '_default'
-------------------------------------------
Defaccess          : R
Acls               :
    Accessor                 Access
    Redbook_Authors(GROUP  ) R

Audit mode         : None
Owner              : tmesec
Update time        : 03-Jun-1997 14:25
Updated by         : tmesec

Data for PROCESS '_default'
-------------------------------------------
Defaccess          : R
Acls               :
    Accessor                 Access
    Redbook_Authors(GROUP  ) None

Audit mode         : None
Owner              : tmesec
Update time        : 03-Jun-1997 13:55
Updated by         : tmesec
```

5.16.4.6 Example 6: Security Resource with Resource Type CONNECT

Create a security resource as follows :

Endpoint Type	UNIX
Resource Type	CONNECT
Resource name	rl0230a.itsc.austin.ibm.com
Default Access	None

The command generated at distribution is as follows:

```
newres CONNECT rl0230a.itsc.austin.ibm.com audit(failure) defaccess(none)
  restrictions(days(anyday) time(anytime)) owner (tmesec)
```

Leave all other attributes as default. After distribution, you have these entries in TACF:

```
sr CONNECT rl0230a.itsc.austin.ibm.com
Data for CONNECT 'rl0230a.itsc.austin.ibm.com'
-----------------------------------------------------------
Defaccess        : None  <-----------  This will prevent all groups
Audit mode       : Failure             and users from accessing this
Owner            : tmesec              system.
Create time      : 16-May-1997 11:57
Update time      : 16-May-1997 11:57
Updated by       : tmesec
```

Add this security resource to the security role Write_Redbook:

Resource Access Rights	Read

The command generated at distribution is as follows:

```
authorize CONNECT rl0230a.itsc.austin.ibm.com gid('Redbook_Authors')
  access('Read')
```

The TACF entry now shows:

```
sr CONNECT rl0230a.itsc.austin.ibm.com
Data for CONNECT 'rl0230a.itsc.austin.ibm.com'
-----------------------------------------------------------
Defaccess        : None
Acls             :
    Accessor               Access
    Redbook_Authors(GROUP  ) R

Audit mode       : Failure
Owner            : tmesec
Create time      : 16-May-1997 11:57
Update time      : 21-May-1997 11:42
Updated by       : tmesec
```

User lynn is now able to connect to rl0230a.itsc.austin.ibm.com. Other users/groups who try to connect are refused access because the default access to the resource is still None.

5.16.5 TACF Initialization File

In our installation, we specified directory /usr/local/Tivoli as the location for TACF directories and files. So, the TACF initialization file can be found in /usr/local/Tivoli/TACF/seos.ini.

If you specify a different directory when you install TACF, use that name in place of /usr/local/Tivoli.

The initialization file seos.ini defines the environment in which TACF will start. You can find a full description in the *TME 10 Security Management TACF Reference*, but here is some of the information that it contains. Replace the directory name with the directory of your installation:

- Installation directory

  Default: /usr/local/Tivoli/TACF

- Directory of the seadm rulers and other configuration files

  Default: /usr/local/Tivoli/TACF/data

- TACF database

  Default: /usr/local/Tivoli/TACF/seosdb

- TACF log files

  Default: /usr/local/Tivoli/TACF/log/seosd.audit
  /usr/local/Tivoli/TACF/log/seosd.error
  /usr/local/Tivoli/TACF/log/seosd.trace

- Startup file for TACF Daemons

  Default : /usr/local/Tivoli/TACF/rc.SeOS.base

- Symbolic link to TACF directory

  Default : /usr/seos -> /usr/local/Tivoli/TACF

- Options for TACF trace messages
- Options for resolving IP addresses to hostnames

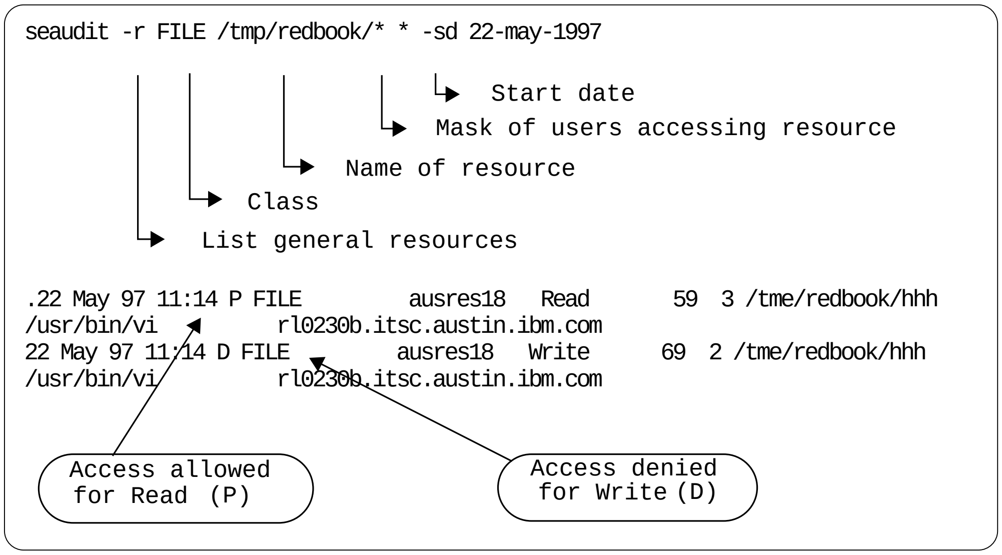
5.16.6 TACF Auditing

The location of the audit file is defined in the initialization file seos.ini and the default is /usr/local/Tivoli/TACF/log/seosd.audit.

Using our example of resource /tme/redbook/*, set up as follows:

- Default access Read
- ACLs include group Redbook_Authors, where lynn is a member of the group
- Audit mode set to Success and Failure

From user ID ausres18, we edit /tme/redbook/hhh and try to save new data. This is what we get from the audit log with the command:



```
seaudit -r FILE /tmp/redbook/* * -sd 22-may-1997

.22 May 97 11:14 P FILE        ausres18   Read       59  3 /tme/redbook/hhh
/usr/bin/vi             rl0230b.itsc.austin.ibm.com
22 May 97 11:14 D FILE        ausres18   Write      69  2 /tme/redbook/hhh
/usr/bin/vi             rl0230b.itsc.austin.ibm.com
```

Start date
Mask of users accessing resource
Name of resource
Class
List general resources
Access allowed for Read (P)
Access denied for Write (D)

The default access to the resource allows ausres18 Read access to the resource, but since the user ID is not defined in the ACLs, Write access is denied.

If the audit mode of /tme/redbook/* is set to Failure only, successful accesses are not logged.

5.16.7 TACF Problem Determination

There are several tools available to help understand what TACF is doing and help you find where things might be going wrong:

5.16.7.1 TACF Command Line Interface

We have already looked at the TACF Command Line in “TACF Commands” on page 112. The TACF CLI can be used to check at the database level that resources and access permissions are defined as expected.

5.16.7.2 TACF Trace

TACF tracing can be controlled through the initialization file seos.ini. Here are the relevant lines from seos.ini:

```
; The destination of trace messages. There are several valid options:
; file       - Trace messages are written to a file.
; none       - Trace messages are not written at all.
; file,stop  - Trace messages are written to a file and automatically
;              disabled once the daemon has passed it's initialization.
; Default Value: file,stop
trace_to = file


; Location of the file to which trace messages are written
; Default Value: /usr/seos/log/seosd.trace
trace_file = /usr/local/Tivoli/TACF/log/seosd.trace


; Trace file type. This can be one of the following values:
; text   - The trace messages file is a text file.
; binary - The trace messages file is a binary file which
;          reduces the size required by this file.
; Default Value: text
trace_file_type = text

; Location of a trace filtering file.
; Default value: /usr/local/Tivoli/TACF/etc/trcfilter.
init
trace_filter = /usr/local/Tivoli/TACF/etc/trcfilter.in
it

; Free space to leave in filesystem when trace_to file is allowed.
; When less than this space is free, trace will be disabled (in Kbytes)
; Default Value: 1024 (1MB)
trace_space_saver = 5120
```

To start a full trace, set:

```
trace_to = file
```

or from user ID tmesec or from a user with the ADMIN attribute, enter:

```
secons -t+
```

To watch the trace as resources are accessed:

```
tail -f /usr/local/Tivoli/TACF/log/seosd.trace
or
secons -tv
```

To check the status of the trace:

```
secons -ts
```

The trace is automatically disabled if free space in the filesystem gets too low.

5.16.7.3 Examples of TACF Traces

Here are some examples of what you might see based on the examples we showed earlier in “Examples of TME Security Records Mapped onto TACF” on page 117.

At the end of “Example 3: Create a SecSurity Role” on page 122, user lynn (uid 202) has RW access to /tme/redbook/* through the security role Redbook_Authors. Here are the trace messages that are generated when lynn tries to edit /tme/redbook/hhh.

```
21 May 97 09:46:11> FORK    : P=19038 U=202  G=1    Child=19302 Pgm:/usr/bin/ksh
21 May 97 09:46:11> EXEC    : P=19302 U=202  G=1    (D=a0007    I=12624 )
Pgm:/usr/bin/vi
21 May 97 09:46:11> EXECARGS: 'vi hhh'
21 May 97 09:46:11> FILE    : P=19302 (/usr/bin/vi) U=202  (D=a0009    I=2049  )
    READ,  :/tme/redbook/hhh
21 May 97 09:46:11> FILE    > (/usr/bin/vi) Result: 'P' [stage=57 gstag=57
ACEEH=6 rv=0(/tme/redbook/*)]
                Why?    Resource ACL check for user groups
21 May 97 09:46:36> FILE    : P=19302 (/usr/bin/vi) U=202  (D=a0009    I=2049  )
   WRITE,  :/tme/redbook/hhh
21 May 97 09:46:36> FILE    > (/usr/bin/vi) Result: 'P' [stage=57 gstag=57
ACEEH=6    rv=0(/tme/redbook/*)]
                Why?    Resource ACL check for user groups
```

This can be deciphered as:

- User lynn (U=202) executes command `vi hhh`.

  Remember that if you have issued an `su` command to change to another user ID, the UID that you see here is still the original UID, not the one you have `su`'d to. You have access only to the recourses available to the original UID.
- The EXEC message indicates that TACF received an execution request, but because the program is not setuid or setgid and is not defined as a resource, TACF grants execution without any further checking of default access rules.
- TACF checks for an entry for FILE /tme/redbook/hhh to see if lynn is authorized to READ the file.
- TACF allows access (Result: 'P') to /tme/redbook/* when it finds lynn is authorized because the user ID is a member of group Redbook_Authors.
- TACF tells us the reason it allows access in the *Why?* message “resource ACL check for user groups”. Authorization has been granted because lynn is a member of group Redbook_Authors, and this group is in the ACL list of `/tme/redbook/*` with access levels R and W.
- TACF checks again for WRITE access when the file is saved and again allows access.

After “Example 6: Security Resource with Resource Type CONNECT” on page 130, security resource rl0230a.itsc.austin.ibm.com is defined with default access None and Read access for group Redbook_Authors. Here are the messages that are generated when user lynn (UID 202) and user root (UID 0) try to access this system.

```
21 May 97 09:57:02> FORK    : P=21578 U=0 G=1 Child=20192 Pgm:/usr/bin/ksh
21 May 97 09:57:02> EXECsu  : P=20192 U=0    G=1    (D=a0007    I=12828 )
Pgm:/usr/bin/tn
21 May 97 09:57:02> EXECARGS: 'tn rl0230a'
21 May 97 09:57:02> EXEC    > Result: 'P' [stage=1059 gstag=1059 ACEEH=2    rv=0
(/usr/bin/tn)
                Why?    Default record universal access check
21 May 97 09:57:02> SUID >P=20192 U=0(R=0 E=0 S=0 ) to (R=0 E=0 S=-1 ) () BYPASS
21 May 97 09:57:02> CONNECT : P=20192 U=0    ACEEH=2    from 9.3.1.113:23   to
socket 0     host=rl0230a.itsc.austin.ibm.com
21 May 97 09:57:02> CONNECT > Result 'D' P=20192 ACEEH=2
TERM=rl0230a.itsc.austin.ibm.com
                Why?    No rule granting access to resource


21 May 97 11:43:05> FORK    : P=19150 U=202  G=1    Child=24720 Pgm:/usr/bin/ksh
21 May 97 11:43:05> EXECsu  : P=24720 U=202  G=1    (D=a0007 I=12828 )
Pgm:/usr/bin/tn
21 May 97 11:43:05> EXECARGS: 'tn rl0230a'
21 May 97 11:43:05> EXEC > Result: 'P' [stage=1059 gstag=1059 ACEEH=6
rv=0(/usr/bin/tn)
                Why?    Default record universal access check
21 May 97 11:43:05> SUID    > P=24720 U=202  (R=202  E=0    S=0   ) to
(R=202  E=202  S=-1  ) () BYPASS
21 May 97 11:43:05> CONNECT : P=24720 U=202  ACEEH=6    from 9.3.1.113:23   to s
ocket 0     host=rl0230a.itsc.austin.ibm.com
21 May 97 11:43:05>CONNECT >Result:'P'P=24720 ACEEH=6 TERM=rl0230a.itsc.austin.ibm
.com
                Why?    Resource ACL check for user groups
```

The messages mean:

- EXEC su recognizes `/usr/bin/tn` as a setuid program.
- Access is granted to `/usr/bin/tn` by the default access level set in the _default record in class PROGRAM.
- SUID with BYPASS means that TACF decided it did not need to do an authorization check on the SURROGATE access rule. This can be for several reasons:
  - A setuid request was to the same UID as the current UID.
  - The program is a privileged program as listed in /usr/local/Tivoli/TACF/etc/privpgms.init.
  - The program is a login program as listed in /usr/local/Tivoli/TACF/etc/loginpgms.init.
- root (U=0) is denied access (Result ’D’) to rl0230a.itsc.austin.ibm.com because the default access is None and root is not in the ACLs of this resource.
- lynn (U=202) is allowed access (Result ’P’) because it is in the ACLs of the resource.

5.16.7.4 TACF Error Log

The location of the TACF error log is defined in the initialization file seos.ini. The default location is /usr/local/Tivoli/TACF/log/seos.error.

To display the errors from the error log, issue:

```
seerrlog
```

In our experimentation, we did not encounter any situations that resulted in messages being entered into the TACF error log. However, if other sources of messages are not helping you to resolve a problem, then it may be worth checking this log.

Chapter 6. Installing TME 10 Security Management

This chapter describes some of the considerations for installing TME 10 Security Management and includes an example of the installation process. Refer to the *TME 10 Security Management User's Guide* "Installing TME 10 Security Management" chapter for further information regarding installation.

6.1 Software Requirements

As a prerequisite you must have TME 10 Framework at least at Version 3.1 installed and running. If using TME 10 Framework at Version 3.1, some additional patches will be required. For additional information regarding required patches, please refer to *TME 10 Security Management Release Notes*. These notes are provided in the product packaging. You should also refer to the release notes to check for any changes to the list of supported platforms given below.

This is a list of the management platforms supported in the first release of TME 10 Security Management; the same platforms are also supported as managed nodes:

- AIX 4.1.x and 4.2.x
- HP-UX 9.x and 10.x
- Windows NT Server or Workstation 3.x and 4.x
- Sun Solaris > 2.3
- Sun OS 4.1.2 and 4.1.3

6.2 Hardware Requirements

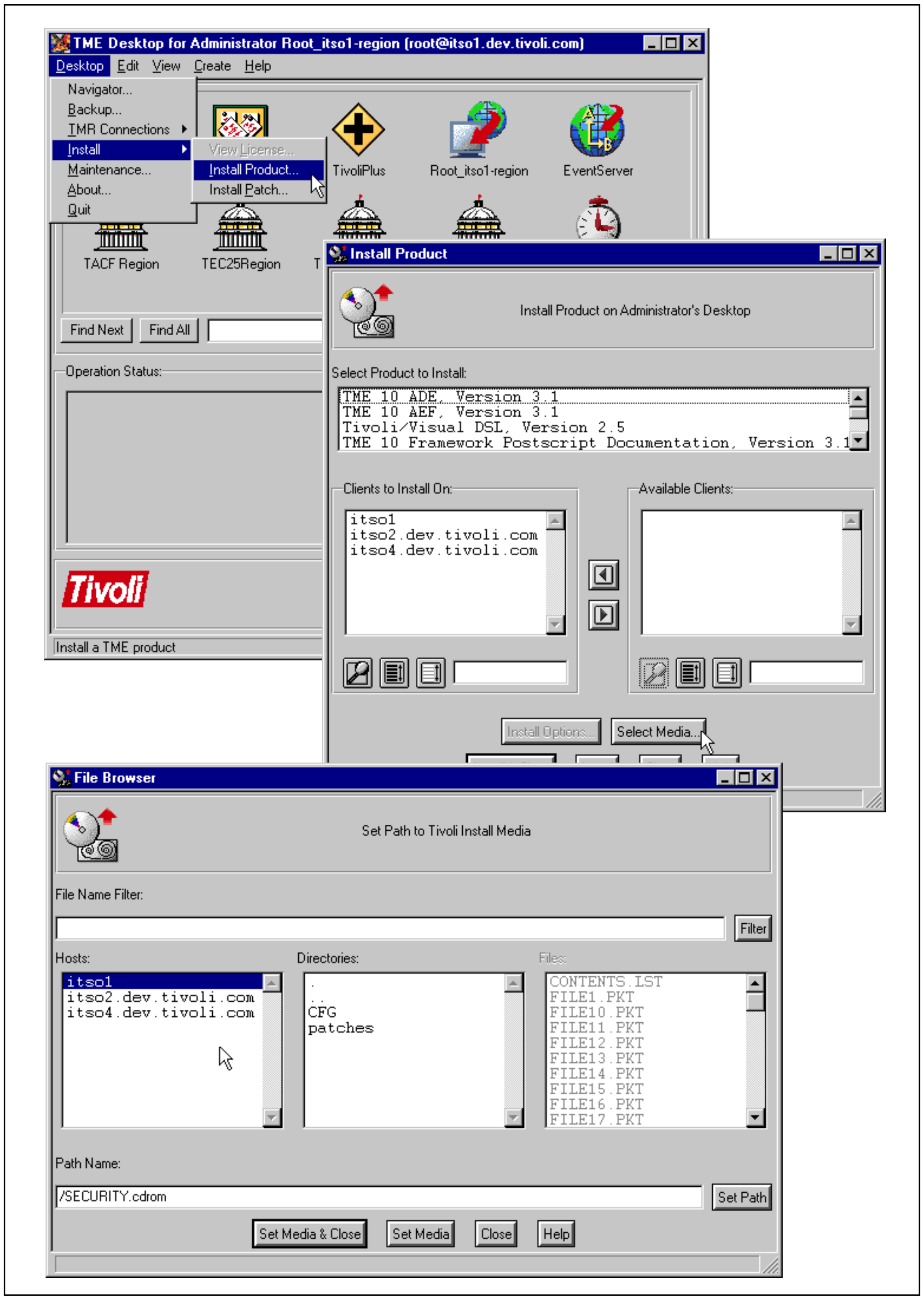
Refer to the *TME 10 Security Management Release Notes* for client and server disk space requirements for this application.

6.3 Installing TME 10 Security Management

You can install tje TME 10 Security Management application from the TME desktop or from the command line.

To install the product, you must have the *install_product* authorization role,

The installation process for TME 10 Security Management is much the same as it is for most TME 10 applications. The following is a run-through of an installation. Refer also to the *TME 10 Security Management Users Guide* supplied with the product for further information on the install process or for details of how to perform the install from the command line. `winstall` can be used to install all TME 10 Security Management components.



Select Product to Install:

```
TME 10 ADE, Version 3.1
TME 10 AEF, Version 3.1
Tivoli/Visual DSL, Version 2.5
TME 10 Framework Postscript Documentation, Version 3.1
```

Clients to Install On:

```
itso1
itso2.dev.tivoli.com
itso4.dev.tivoli.com
```

Hosts:

```
itso1
itso2.dev.tivoli.com
itso4.dev.tivoli.com
```

Directories:

```
.
..
CFG
patches
```

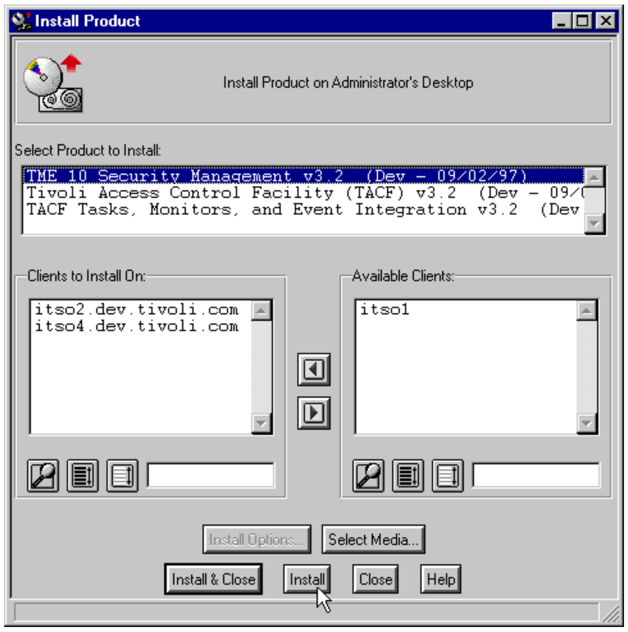
Files:

```
CONTENTS.LST
FILE1.PKT
FILE10.PKT
FILE11.PKT
FILE12.PKT
FILE13.PKT
FILE14.PKT
FILE15.PKT
FILE16.PKT
FILE17.PKT
```

Path Name: /SECURITY.cdrom

Figure 39. Initiate TME 10 Product Install Process

A Tivoli administrator with the correct authority initiates the **Install Product** option from the Desktop menu, Install submenu. If the previous install media path is incorrect, then you must click on **Select Media...** to change the install path to point to the source of the TME 10 Security Management product. Select **Set Media & Close** to return to the installation process.



Select Product to Install:

```
TME 10 Security Management v3.2   (Dev - 09/02/97)
Tivoli Access Control Facility (TACF) v3.2   (Dev - 09/(
TACF Tasks, Monitors, and Event Integration v3.2   (Dev
```

Clients to Install On:

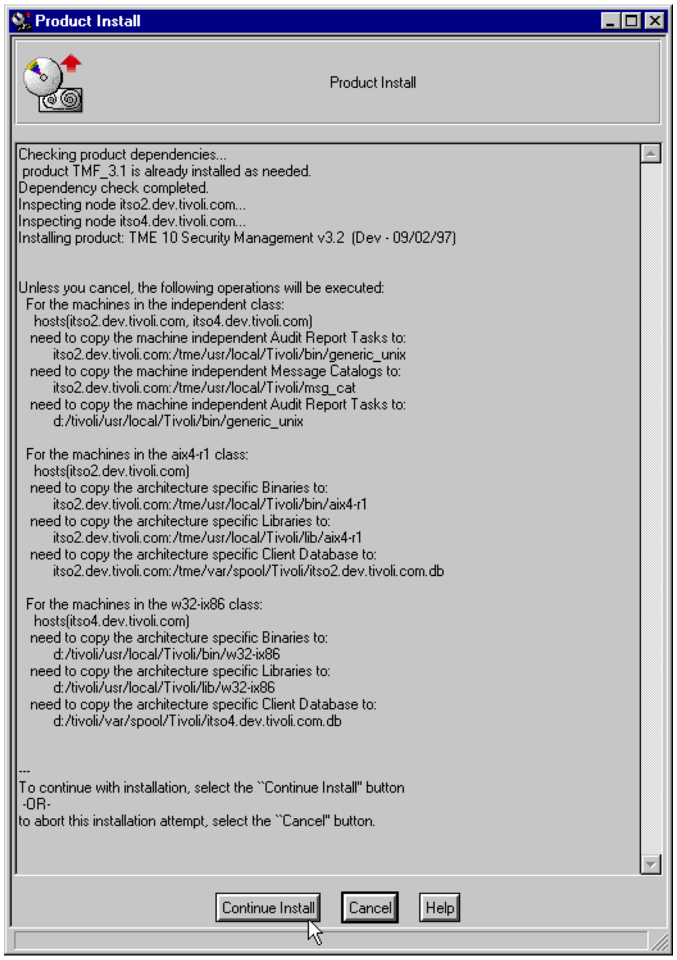
```
itso2.dev.tivoli.com
itso4.dev.tivoli.com
```

Available Clients:

```
itso1
```

Figure 40. Selecting Product to Install

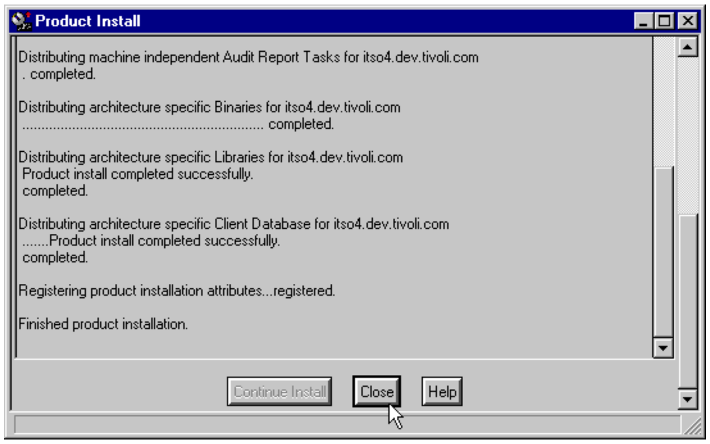
The order in which you install the TME 10 Security Management pieces is not significant. Note that the Tivoli Access Control Facility (TACF) will only install on supported UNIX platforms—it is not required and will not install on NT. Here in Figure 40 we choose to install the TME 10 Security Management base product. From the list of available clients, we can then select the systems on which we wish to install the product. In this example, itso1 is the TMR server and already has the product installed; itso2 is an AIX 4.1 system, and itso4 is an NT server. By default, the install process lists all servers and endpoints in the current TMR that do not already have the product installed in the *Clients to install on* list.



```
Checking product dependencies...
 product TMF_3.1 is already installed as needed.
Dependency check completed.
Inspecting node itso2.dev.tivoli.com...
Inspecting node itso4.dev.tivoli.com...
Installing product: TME 10 Security Management v3.2  (Dev - 09/02/97)


Unless you cancel, the following operations will be executed:
  For the machines in the independent class:
    hosts(itso2.dev.tivoli.com, itso4.dev.tivoli.com)
   need to copy the machine independent Audit Report Tasks to:
        itso2.dev.tivoli.com:/tme/usr/local/Tivoli/bin/generic_unix
   need to copy the machine independent Message Catalogs to:
        itso2.dev.tivoli.com:/tme/usr/local/Tivoli/msg_cat
   need to copy the machine independent Audit Report Tasks to:
        d:/tivoli/usr/local/Tivoli/bin/generic_unix

  For the machines in the aix4-r1 class:
    hosts(itso2.dev.tivoli.com)
   need to copy the architecture specific Binaries to:
        itso2.dev.tivoli.com:/tme/usr/local/Tivoli/bin/aix4-r1
   need to copy the architecture specific Libraries to:
        itso2.dev.tivoli.com:/tme/usr/local/Tivoli/lib/aix4-r1
   need to copy the architecture specific Client Database to:
        itso2.dev.tivoli.com:/tme/var/spool/Tivoli/itso2.dev.tivoli.com.db

  For the machines in the w32-ix86 class:
    hosts(itso4.dev.tivoli.com)
   need to copy the architecture specific Binaries to:
        d:/tivoli/usr/local/Tivoli/bin/w32-ix86
   need to copy the architecture specific Libraries to:
        d:/tivoli/usr/local/Tivoli/lib/w32-ix86
   need to copy the architecture specific Client Database to:
        d:/tivoli/var/spool/Tivoli/itso4.dev.tivoli.com.db


---
To continue with installation, select the ``Continue Install'' button
 -OR-
to abort this installation attempt, select the ``Cancel'' button.
```

Figure 41. Product Install Information Dialog

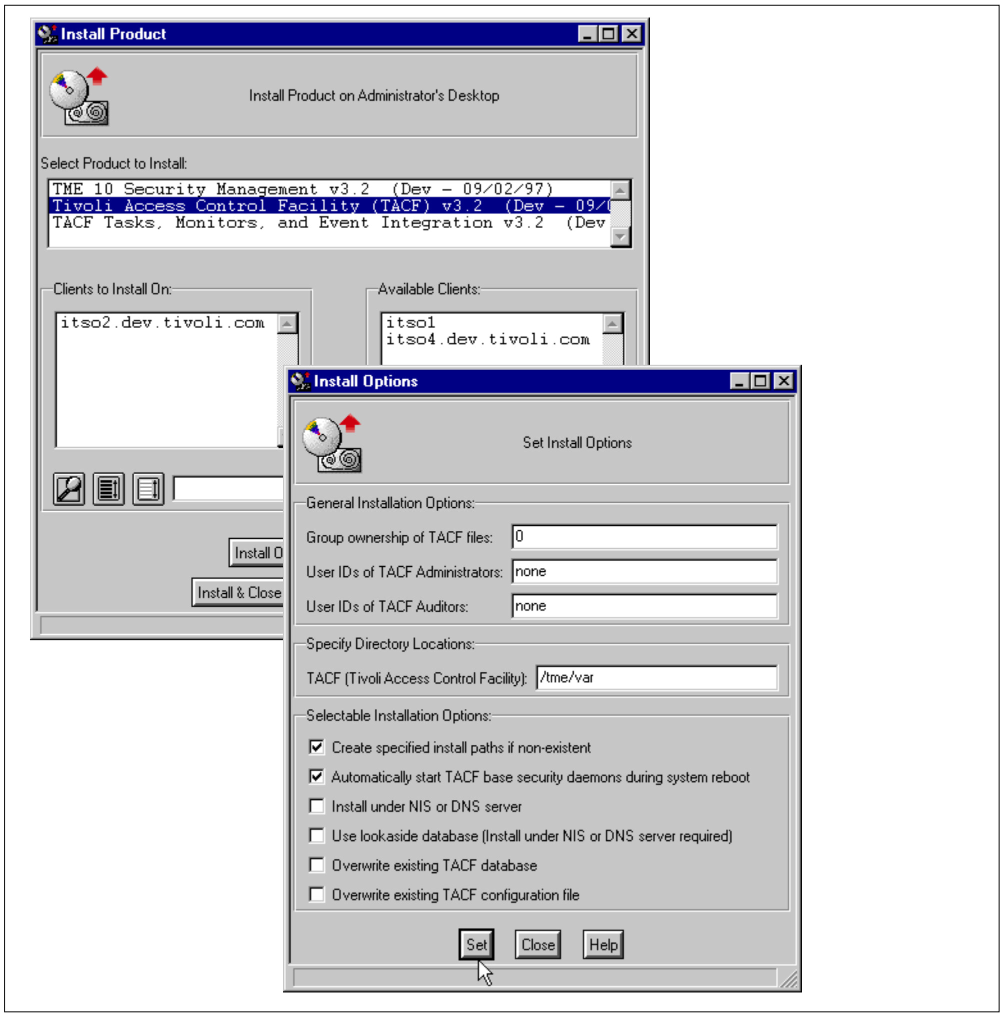
The install process checks the target machines. If the information dialog contains no errors, it lists what is to be done for each class (platform type) and the hostnames of the machines in that class. Now we can select **Continue Install** to proceed.



```
Distributing machine independent Audit Report Tasks for itso4.dev.tivoli.com
 . completed.

Distributing architecture specific Binaries for itso4.dev.tivoli.com
 ............................................................ completed.

Distributing architecture specific Libraries for itso4.dev.tivoli.com
 Product install completed successfully.
 completed.

Distributing architecture specific Client Database for itso4.dev.tivoli.com
 .......Product install completed successfully.
 completed.

Registering product installation attributes...registered.

Finished product installation.
```

Figure 42. Finished Product Installation Messages Dialog

Check the messages to ensure the install completed and select **Close** to exit the dialog.


Install Product
Install Product on Administrator's Desktop
Select Product to Install:

```
TME 10 Security Management v3.2  (Dev - 09/02/97)
Tivoli Access Control Facility (TACF) v3.2  (Dev - 09/(
TACF Tasks, Monitors, and Event Integration v3.2  (Dev
```

Group ownership of TACF files: 0
User IDs of TACF Administrators: none
User IDs of TACF Auditors: none

Specify Directory Locations:
TACF (Tivoli Access Control Facility): /tme/var

Selectable Installation Options:
- [x] Create specified install paths if non-existent
- [x] Automatically start TACF base security daemons during system reboot
- [ ] Install under NIS or DNS server
- [ ] Use lookaside database (Install under NIS or DNS server required)
- [ ] Overwrite existing TACF database
- [ ] Overwrite existing TACF configuration file

Figure 43. Installing TACF

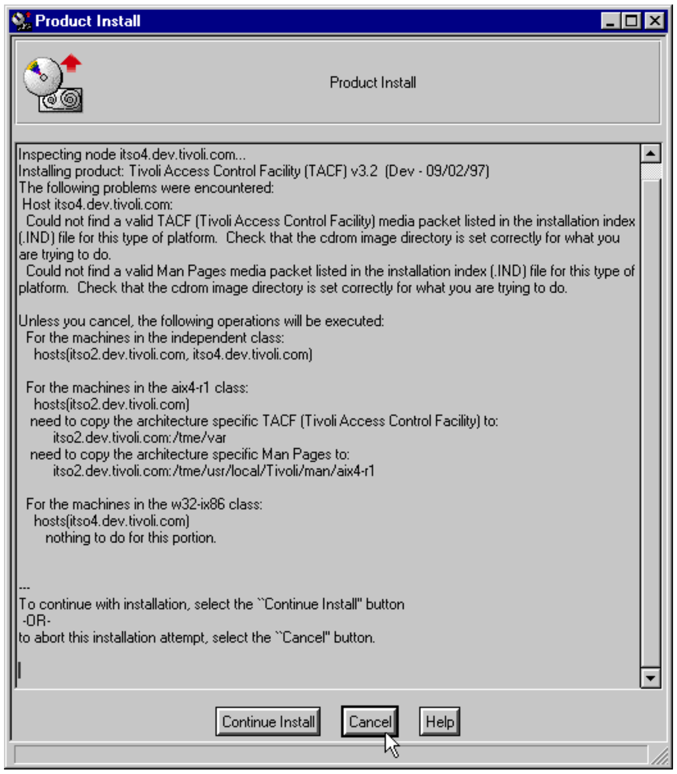
When you select to install TACF, you are asked to specify Install Options. If you dismiss the options dialog before you mean to, you can press the **Install Options** button on the install product dialog to display it again. The options are as follows:

Group Ownership	Specify the group that will own TACF files and binaries. Default of 0 indicates root group on most systems.
TACF admin ID	Specify IDs additional to tmesec that will assume ownership rights of TACF files. Specifying `root` for this ID is not recommended as you will then be unable to restrict `root` user activities.
TACF Directories	Specify where you want TACF to be located.
Create paths	Select to allow installation to create paths that do not exist.
Autostart Daemons	Ensure TACF is started after future reboots.
Install under NIS	Select if one or more managed nodes is an NIS master or DNS server (automatic for Solaris 2.5 or greater.
Use lookaside DB	Read NIS and DNS entries into a TACF lookaside database during install.

Overwrite TACF DB	Overwrite any existing TACF/SeOS database.
Overwrite TACF Conf	Replace any existing seos.ini file.

Solaris Users Note

TACF installation on Sunsoft Solaris platforms requires the target to be restarted in order for TACF to operate. No other platforms have this restriction.

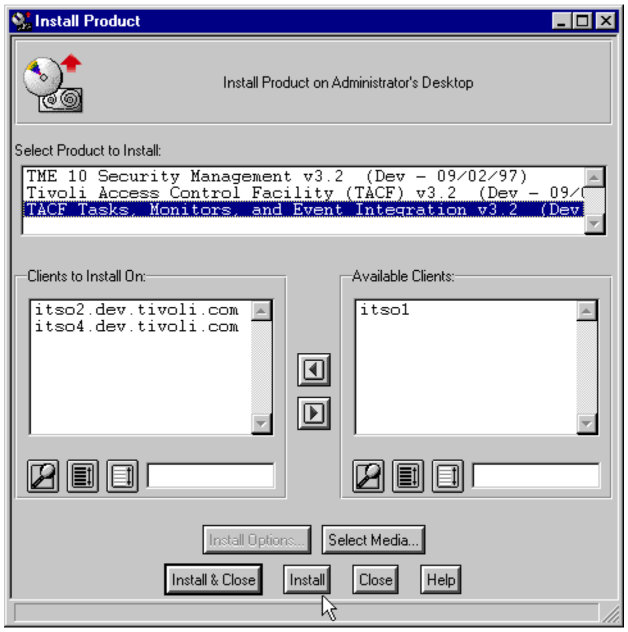


```
Inspecting node itso4.dev.tivoli.com...
Installing product: Tivoli Access Control Facility (TACF) v3.2  (Dev - 09/02/97)
The following problems were encountered:
 Host itso4.dev.tivoli.com:
  Could not find a valid TACF (Tivoli Access Control Facility) media packet listed in the installation index
(.IND) file for this type of platform.  Check that the cdrom image directory is set correctly for what you
are trying to do.
  Could not find a valid Man Pages media packet listed in the installation index (.IND) file for this type of
platform.  Check that the cdrom image directory is set correctly for what you are trying to do.

Unless you cancel, the following operations will be executed:
  For the machines in the independent class:
    hosts(itso2.dev.tivoli.com, itso4.dev.tivoli.com)

  For the machines in the aix4-r1 class:
    hosts(itso2.dev.tivoli.com)
   need to copy the architecture specific TACF (Tivoli Access Control Facility) to:
         itso2.dev.tivoli.com:/tme/var
   need to copy the architecture specific Man Pages to:
         itso2.dev.tivoli.com:/tme/usr/local/Tivoli/man/aix4-r1

  For the machines in the w32-ix86 class:
    hosts(itso4.dev.tivoli.com)
        nothing to do for this portion.


---
To continue with installation, select the ``Continue Install'' button
 -OR-
to abort this installation attempt, select the ``Cancel'' button.
```

Figure 44. Result of Selecting an Unsupported Platform for TACF

If you try to install TACF on an NT system, you will receive messages like those shown in Figure 44. TME 10 cannot (of course!) find any TACF products on the install media to distribute to machines in the w32-ix86 class–Windows NT. You could continue here, and installation will continue on valid machines; or you can return to the installation dialog and remove any NT systems from the install list.

Again, once install has completed, check the messages and close the install message dialog.



Select Product to Install:

```
TME 10 Security Management v3.2  (Dev - 09/02/97)
Tivoli Access Control Facility (TACF) v3.2  (Dev - 09/(
TACF Tasks, Monitors, and Event Integration v3.2  (Dev
```

Clients to Install On:

```
itso2.dev.tivoli.com
itso4.dev.tivoli.com
```

Available Clients:

```
itso1
```

Figure 45. Installing TACF Tasks, Monitors and Event Integration

Lastly, we select to install the TACF tasks, monitors and event integration. Not all of these items are required on all the nodes where we install TME 10 Security Management. This part of the product package includes some items that only make sense if, for example, you have TME 10 Distributed Monitoring installed. If it is not installed, the TACF monitors are not installed.

Note

If you install TME 10 Distributed Monitoring or TME 10 Enterprise Console on a system after installing TME 10 Security Management, you may want to reinstall the TACF Tasks, Monitors and Event Integration option.

After the Tasks, Monitors and Event pieces have completed, we have finished the installation process.

If the install completed successfully, then now would be a good time to take a backup of the TME database so that you can revert to a known, clean installation of TME 10 Security Management if something goes wrong in your initial configuration and setup during the first use of the product.

Chapter 7. Using TME 10 Security Management

This chapter demonstrates some simple implementations of the TME 10 Security Management model of roles, groups and resources to protect various types of objects. The examples are based on specific platforms and resources, but each type of resource control that we demonstrate, of course, applies in many environments.

This discussion assumes you already have policy regions defined in the TME 10 environment.

7.1 TME 10 Security Management Commands

TME 10 Security Management adds a number of commands to the TME environment to manage security profiles from the command line. These are discussed in detail in the *TME 10 Security Management User's Guide,* but we will present some examples here. The command-line examples follow the GUI examples in this chapter. The new commands provided with TME 10 Security Management are:

`wcpsec`	Copies a security profile record to another record
`wcrtsec`	Adds records to a security profile
`wdelsec`	Deletes records from a security profile
`wlssec`	Lists values defined within a record
`wmodsec`	Edits values within an existing record
`wmvsec`	Moves a record from on security profile to another
`wpopsec`	Populates a security profile

If you are experiencing problems with UNIX systems, you might want to work directly with TACF through the TACF command line. This is described in Section 5.16.3, "TACF Commands" on page 112.

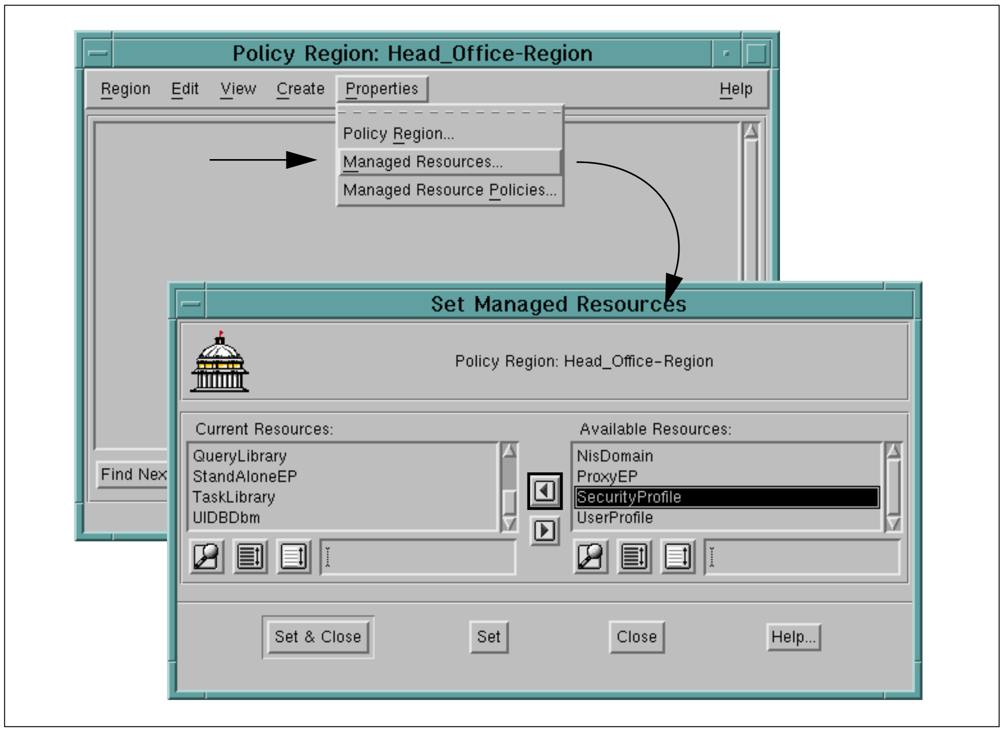
7.2 Sample Security Management Activities

The following activities help to demonstrate exactly how TME 10 Security Management can be used to perform security management tasks.

NOTE: These examples are intended to demonstrate security management techniques and enhance your understanding of issues involved in implementing security management. They have not been exhaustively tested to ensure that they completely provide the desired result.

7.2.1 Create a Security Profile

In order to create a security profile in the profile manager, the managed resources of the policy region must be changed to include the SecurityProfile managed resource type as shown in Figure 46.


Policy Region: Head_Office-Region
Region Edit View Create Properties Help
Policy Region...
Managed Resources...
Managed Resource Policies...
Find Nex
Set Managed Resources

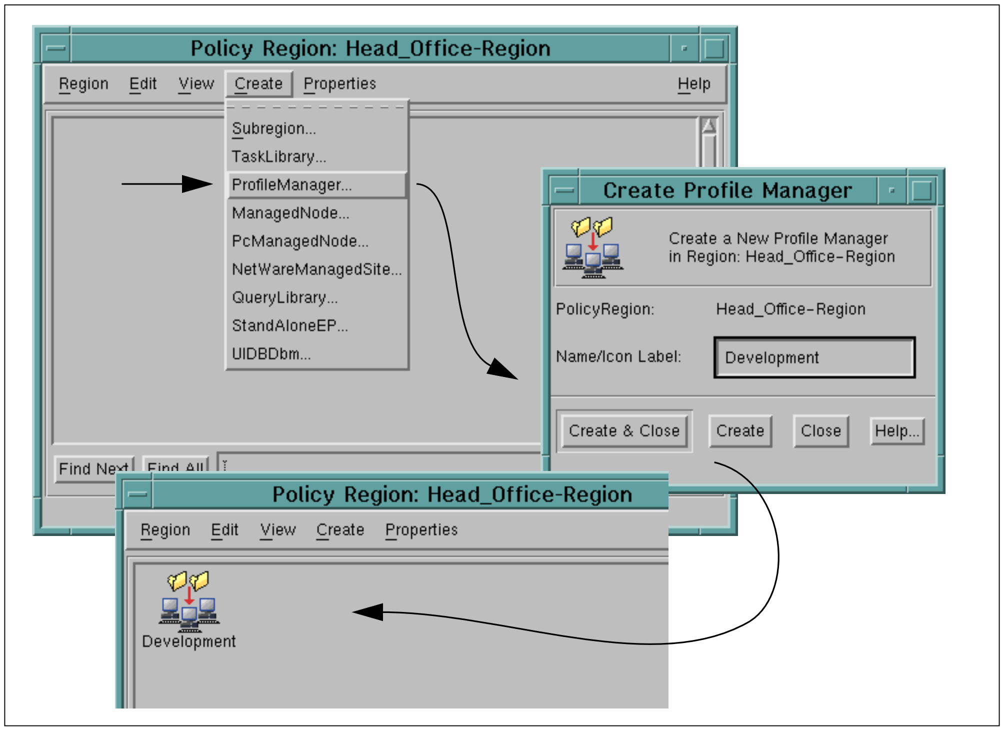
Policy Region: Head_Office-Region

Current Resources:	Available Resources:
QueryLibrary	NisDomain
StandAloneEP	ProxyEP
TaskLibrary	SecurityProfile
UIDBDbm	UserProfile

Set & Close
Set
Close
Help...

Figure 46. Update the Policy Region Managed Resources to Include SecurityProfile

If you do not already have a profile manager in which you wish to create a Security Profile, you need to create one. Figure 47 shows the steps to do this.



Create a New Profile Manager
in Region: Head_Office-Region

PolicyRegion: Head_Office-Region

Name/Icon Label: Development

Create & Close
Create
Close
Help...
Policy Region: Head_Office-Region
Region Edit View Create Properties
Development

Figure 47. Create a Profile Manager

You can then create the security profile by choosing **Profile** from the Create menu, as shown in Figure 48.

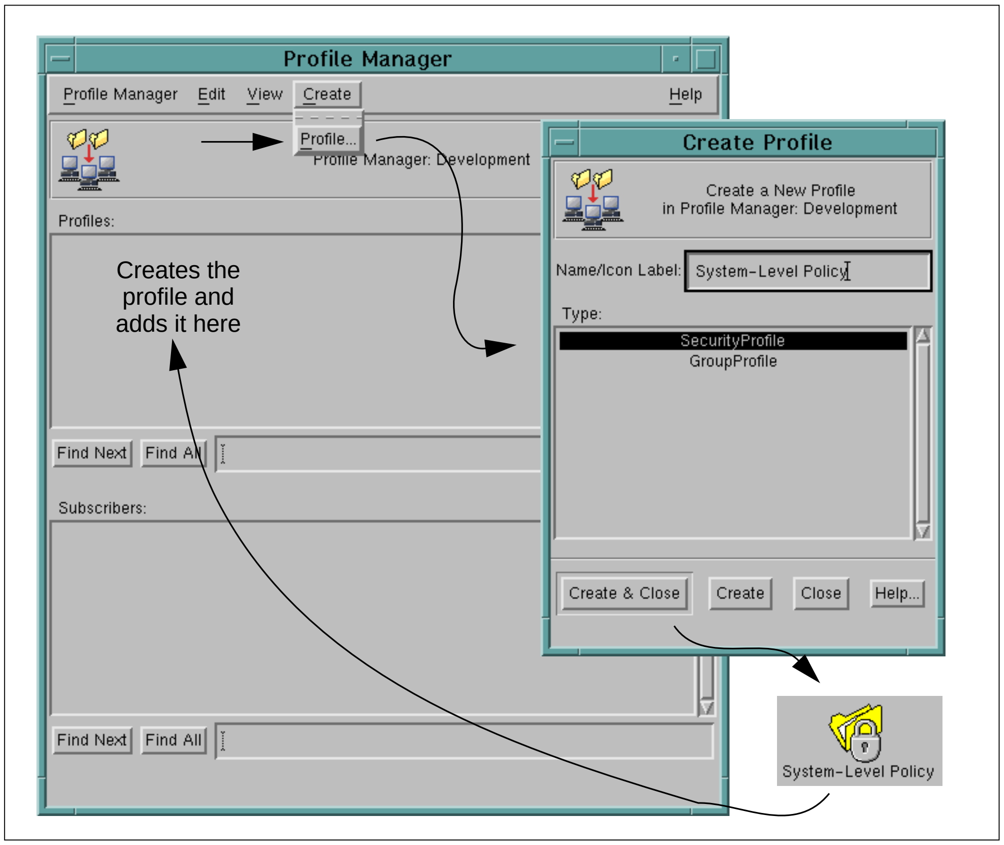


Figure 48. Creating a Security Profile

Now you can start to define the access control of resources from the security profile dialog (Figure 49).

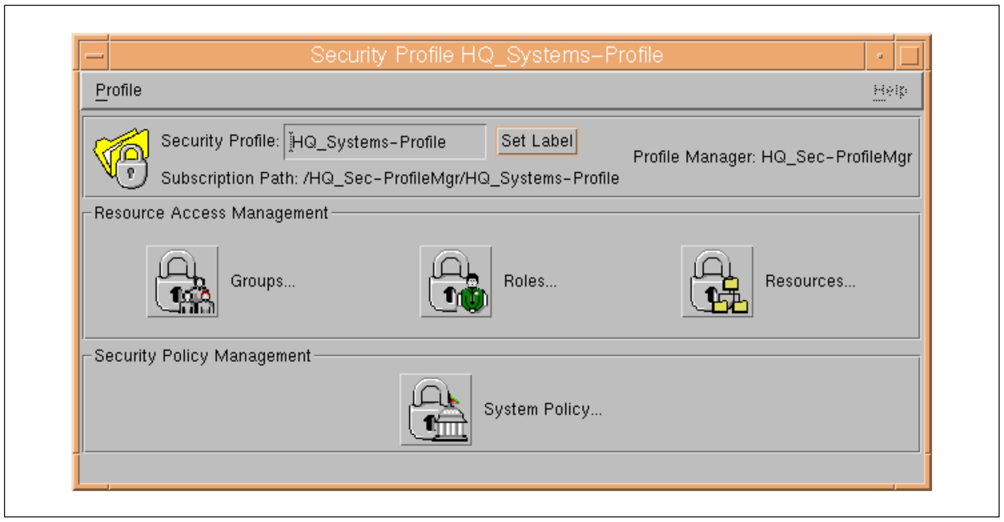

Security Profile HQ_Systems-Profile
Profile
Help
Security Profile: HQ_Systems-Profile
Set Label
Profile Manager: HQ_Sec-ProfileMgr
Subscription Path: /HQ_Sec-ProfileMgr/HQ_Systems-Profile
Resource Access Management
Groups...
Roles...
Resources...
Security Policy Management
System Policy...

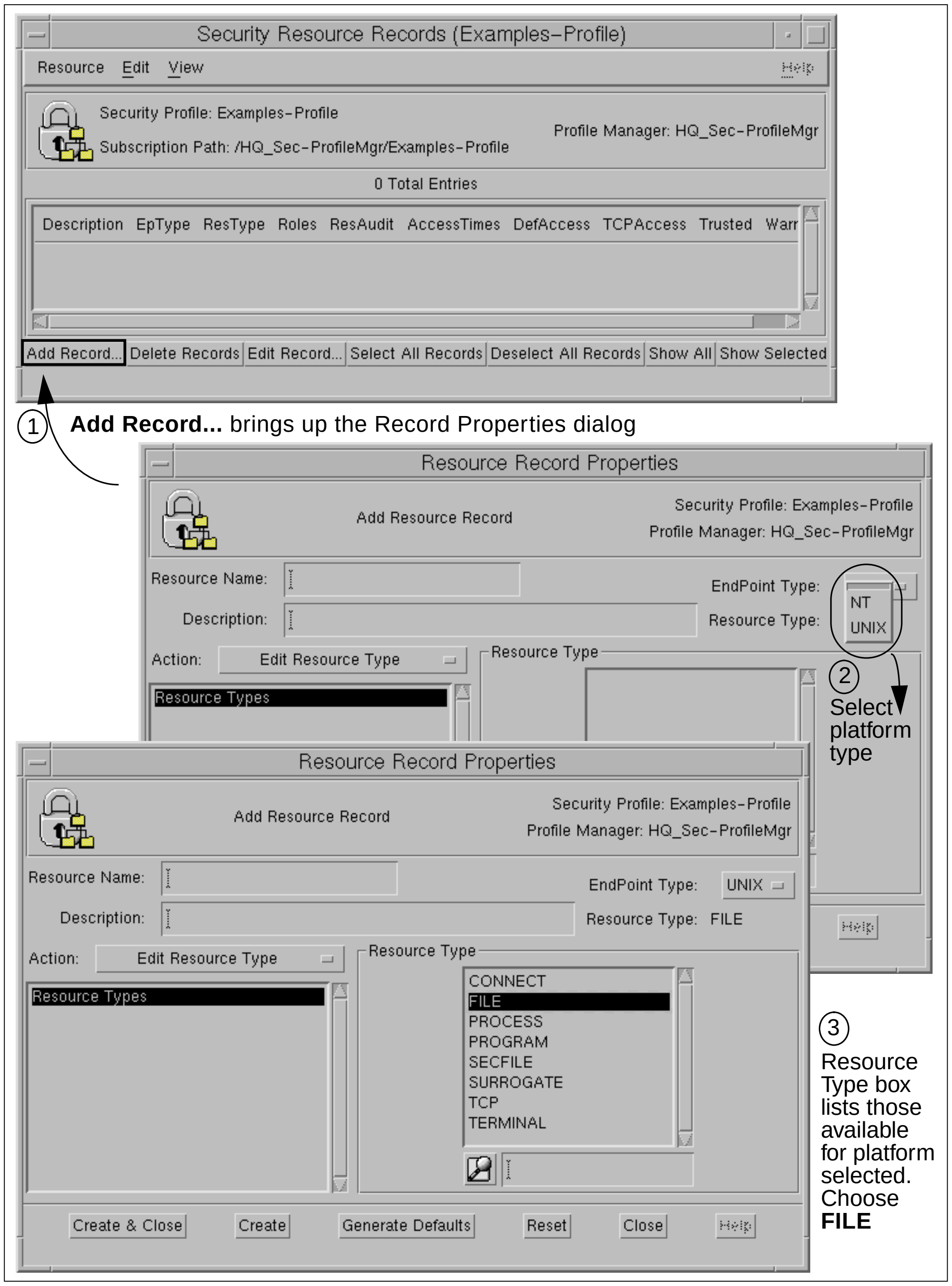
Figure 49. Security Profile Dialog

7.2.2 Restricting User/Group Modifications on Solaris

This example would prevent a root or any undesired user from administering user accounts and modifying group memberships on Sunsoft Solaris. The steps are:

1. Create a security resource record that sets the default access to the user modification commands (usermod, useradd, userdel) to None by specifying the file type /usr/sbin/user* as the resource. On a TACF-installed system, this would prevent anyone, including root, from executing these commands.

   First, we open the resource record dialog from the security profile, add a record and select the platform and resource type to protect (Figure 50).



Security Profile: Examples–Profile
Subscription Path: /HQ_Sec–ProfileMgr/Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr
0 Total Entries

1 **Add Record...** brings up the Record Properties dialog

EndPoint Type: UNIX
Resource Type: FILE

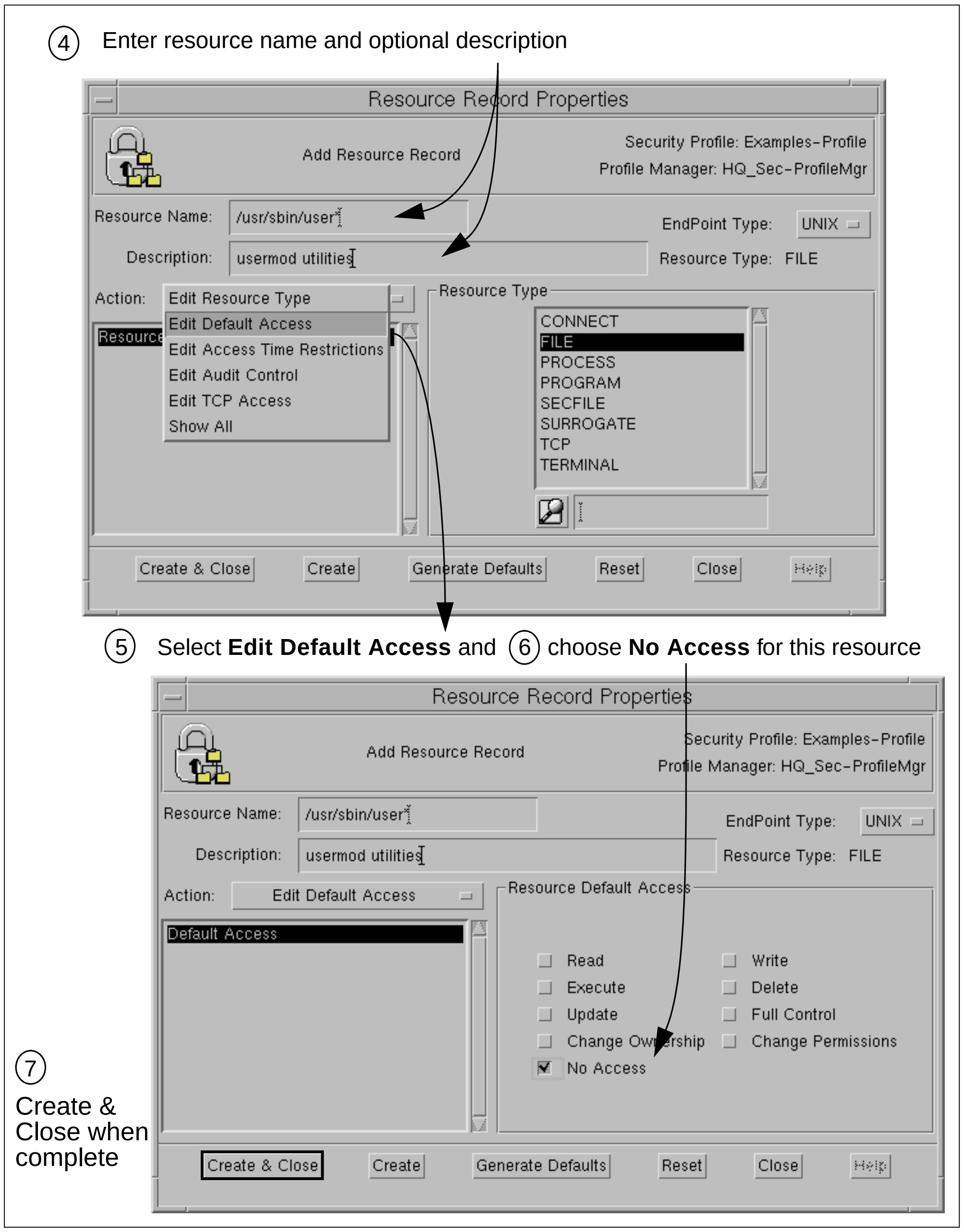
Resource Type list: CONNECT, FILE, PROCESS, PROGRAM, SECFILE, SURROGATE, TCP, TERMINAL

2 Select platform type

3 Resource Type box lists those available for platform selected. Choose **FILE**

Figure 50. Choosing a Platform and Resource Type in a Security Resource Record

The next step is to specify the resource name and edit the default access before creating this new record (Figure 51).



Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr
Resource Name: /usr/sbin/user*
Description: usermod utilities
EndPoint Type: UNIX
Resource Type: FILE
Action: Edit Default Access
[✓ No Access]

Figure 51. Defining Default Access in a Security Resource Record

2. Follow a similar sequence again to create another security resource record that sets the default access to the group modification commands (groupmod, groupadd, groupdel) to None by specifying /usr/bin/group* as the resource (Figure 52).

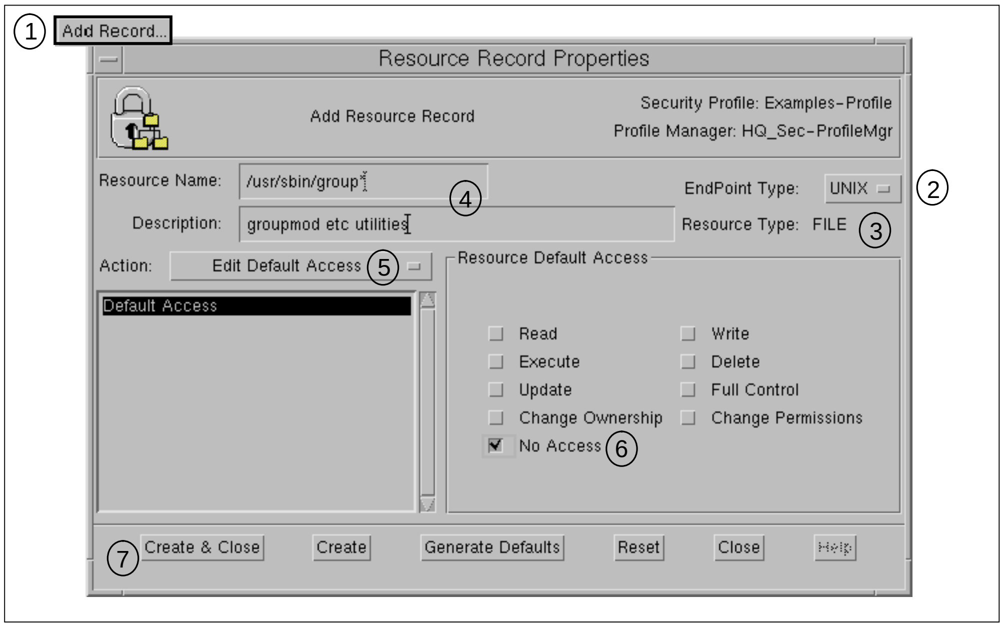


Figure 52. Define Security Resource Record for Solaris Group Utilities

3. Again, we go through a similar sequence to create a third resource record that sets the default access to the Solaris admintool to none (Figure 53).

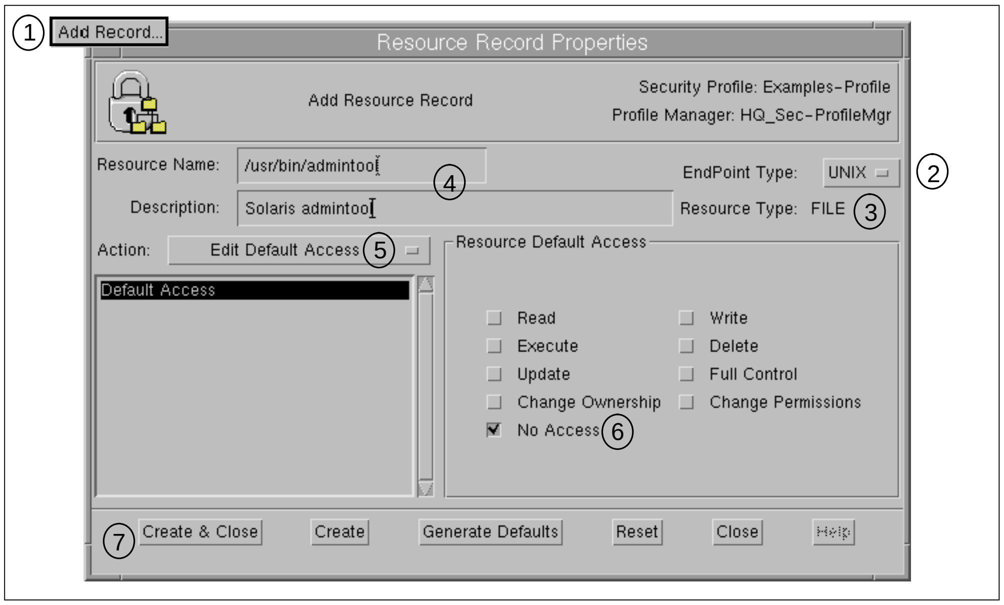


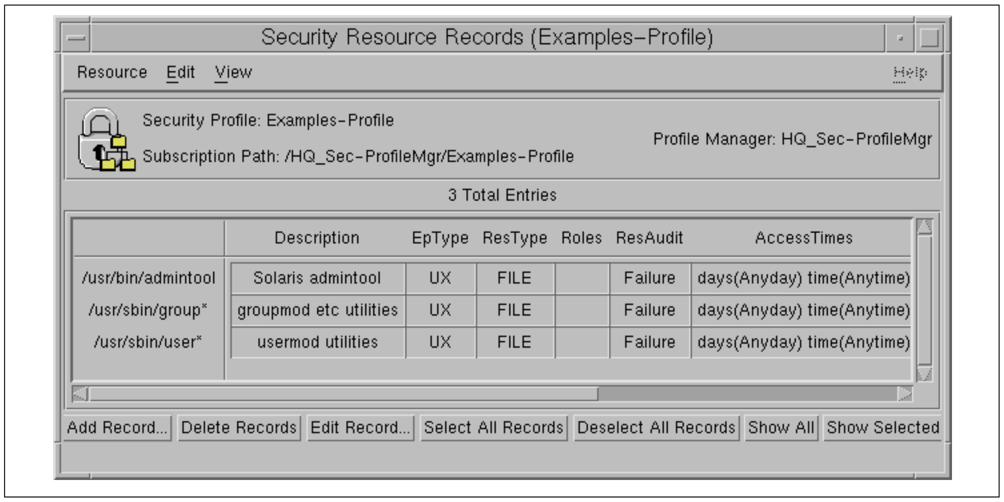
Figure 53. Define Security Resource Record for Solaris Admintool

We now have three security resource records defining a default of No Access for user and group administration utilities.

The TME commands used to produce a similar set of three resources are as follows:

```
wcrtsec Resource -s DefAccess='perms(N)' @Examples-Profile UX:FILE:/usr/sbin/user*
wcrtsec Resource -s DefAccess='perms(N)' @Examples-Profile UX:FILE:/usr/sbin/group*
wcrtsec Resource -s DegAccess='perms(N)' @Examples-Profile UX:FILE:/usr/bin/admintool
```

The resource records dialog for this security profile now lists our three resources. Note that until we assign these resources to a role, the role attribute column is empty (Figure 54).



Security Profile: Examples–Profile
Subscription Path: /HQ_Sec–ProfileMgr/Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr

3 Total Entries

	Description	EpType	ResType	Roles	ResAudit	AccessTimes
/usr/bin/admintool	Solaris admintool	UX	FILE		Failure	days(Anyday) time(Anytime)
/usr/sbin/group*	groupmod etc utilities	UX	FILE		Failure	days(Anyday) time(Anytime)
/usr/sbin/user*	usermod utilities	UX	FILE		Failure	days(Anyday) time(Anytime)

Figure 54. Completed Resource Records Dialog

4. Create a security group called Account Admins whose role is to perform user administration so that we can ensure that someone of our choosing will have access.

   First, we select **Add Record...** from the Group Records dialog, give the group a suitable name and optional description, then we can choose **Edit Member List** to decide who to add to this group (Figure 55).

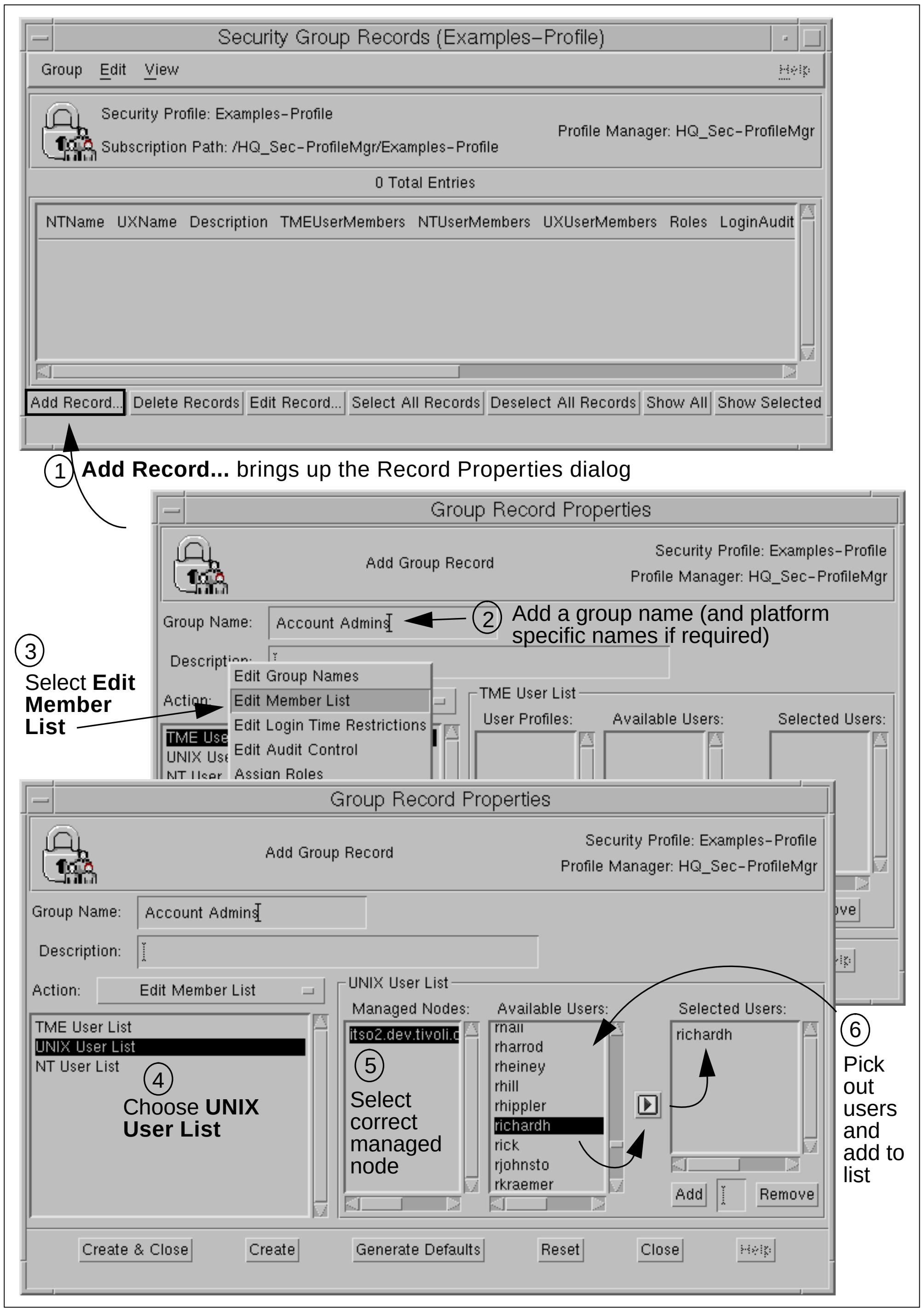

Security Group Records (Examples–Profile)
Group Edit View Help
Security Profile: Examples–Profile
Subscription Path: /HQ_Sec-ProfileMgr/Examples-Profile
Profile Manager: HQ_Sec–ProfileMgr
0 Total Entries
NTName UXName Description TMEUserMembers NTUserMembers UXUserMembers Roles LoginAudit
Add Record... Delete Records Edit Record... Select All Records Deselect All Records Show All Show Selected
1 **Add Record...** brings up the Record Properties dialog
Group Record Properties
Add Group Record
Security Profile: Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr
Group Name: Account Admins
2 Add a group name (and platform specific names if required)
Description:
Action:
Edit Group Names
Edit Member List
Edit Login Time Restrictions
Edit Audit Control
Assign Roles
3 Select **Edit Member List**
TME User List
User Profiles: Available Users: Selected Users:
Group Record Properties
Add Group Record
Security Profile: Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr
Group Name: Account Admins
Description:
Action: Edit Member List
TME User List
UNIX User List
NT User List
4 Choose **UNIX User List**
UNIX User List
Managed Nodes: Available Users: Selected Users:
itso2.dev.tivoli.c
5 Select correct managed node
rhall
rharrod
rheiney
rhill
rhippler
richardh
rick
rjohnsto
rkraemer
richardh
Add Remove
6 Pick out users and add to list
Create & Close Create Generate Defaults Reset Close Help

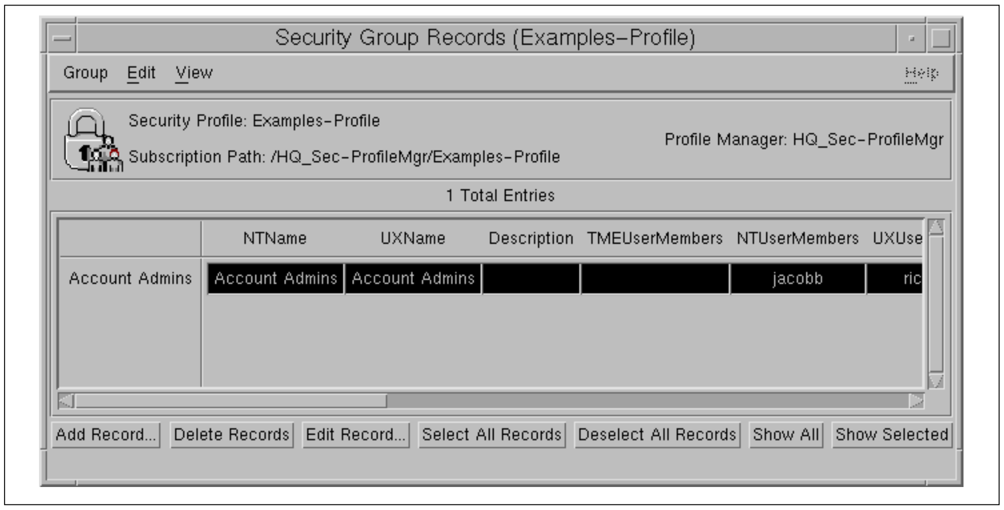
Figure 55. Completing a Security Group Record

We now have a security group containing the user(s) we chose. Note that the users could have come from a number of different sources.

The equivalent TME command to perform this same function looks like this:

```
wcrtsec Group -s UXUserMembers='richardh' @Example-Profile 'Account Admins'
```

The group records dialog now lists the group we created (Figure 56).


Security Group Records (Examples–Profile)
Group Edit View Help

Security Profile: Examples–Profile
Subscription Path: /HQ_Sec–ProfileMgr/Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr

1 Total Entries

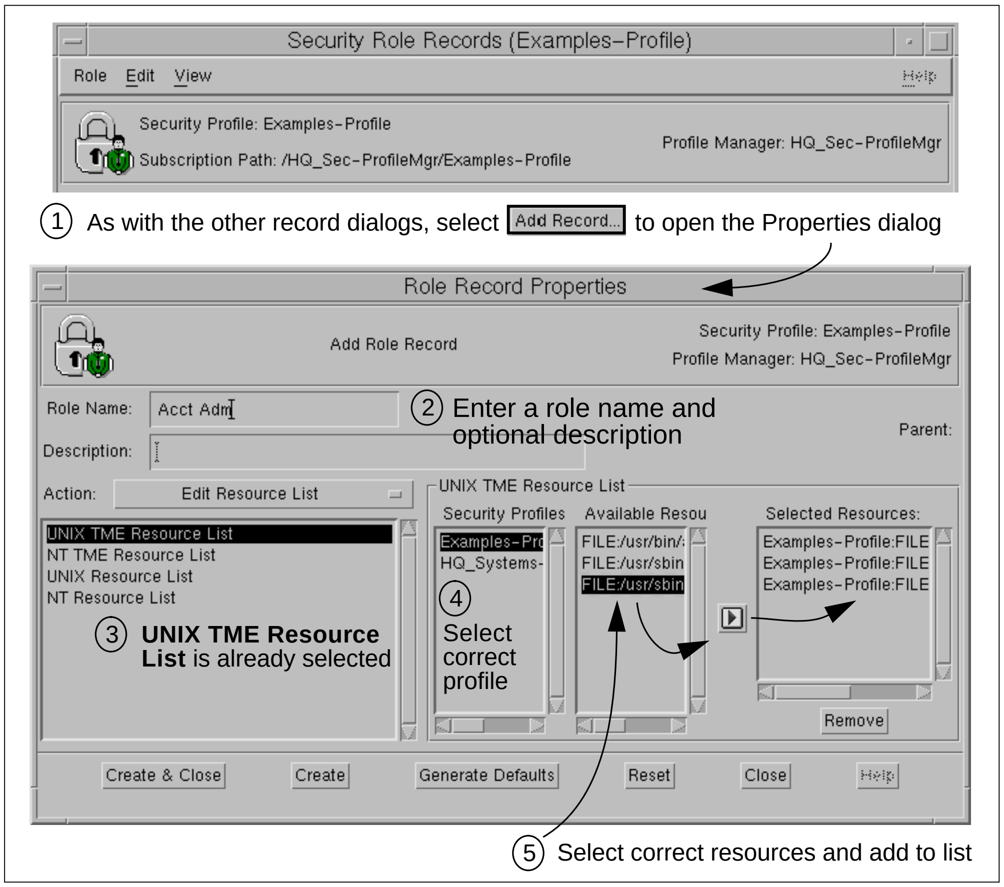
	NTName	UXName	Description	TMEUserMembers	NTUserMembers	UXUse
Account Admins	Account Admins	Account Admins			jacobb	ric

Add Record... Delete Records Edit Record... Select All Records Deselect All Records Show All Show Selected

Figure 56. Completed Group Records Dialog

5. Create a security role called Acct Adm that defines execute access of the previously defined security resources and assign that role to the required security group.

   First, we select **Add Record...** from the Role Records dialog and give the role a name. We can then select resources that will be governed by this role.


Security Role Records (Examples–Profile)
Role Edit View Help
Security Profile: Examples–Profile
Subscription Path: /HQ_Sec–ProfileMgr/Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr
1 As with the other record dialogs, select Add Record... to open the Properties dialog
Role Record Properties
Add Role Record
Security Profile: Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr
2 Enter a role name and optional description
Parent:
Action: Edit Resource List
UNIX TME Resource List
Security Profiles
Available Resou
Selected Resources:
3 UNIX TME Resource List is already selected
4 Select correct profile
Remove
Create & Close Create Generate Defaults Reset Close Help
5 Select correct resources and add to list

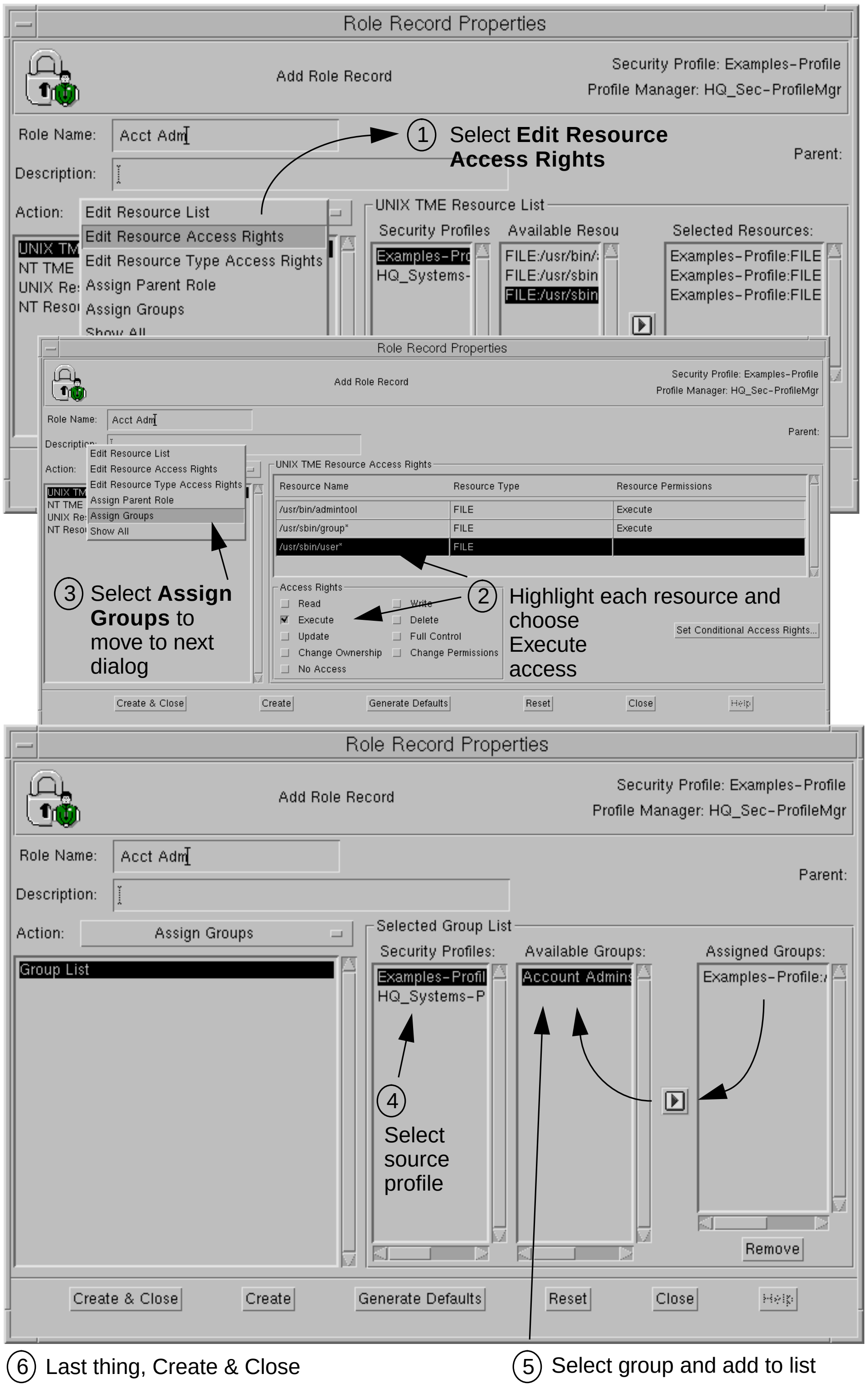
Role Name: Acct Adm
Description:

UNIX TME Resource List
NT TME Resource List
UNIX Resource List
NT Resource List

Security Profiles: Examples–Pro, HQ_Systems–
Available Resources: FILE:/usr/bin/, FILE:/usr/sbin, FILE:/usr/sbin
Selected Resources: Examples–Profile:FILE, Examples–Profile:FILE, Examples–Profile:FILE

Figure 57. Create a Security Role and Add Managed Resources

Next, we specify the rights to the resources we selected, and then we will assign a group whose members are given those rights (Figure 58).


Role Record Properties

Add Role Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Role Name: Acct Adm
Description:
Parent:
Action: Edit Resource List

UNIX TME Resource List

Security Profiles: Examples-Pro, HQ_Systems-
Available Resou: FILE:/usr/bin/, FILE:/usr/sbin, FILE:/usr/sbin
Selected Resources: Examples-Profile:FILE, Examples-Profile:FILE, Examples-Profile:FILE

① Select **Edit Resource Access Rights**

Role Record Properties

Add Role Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Role Name: Acct Adm
Description:
Parent:

UNIX TME Resource Access Rights

Resource Name	Resource Type	Resource Permissions
/usr/bin/admintool	FILE	Execute
/usr/sbin/group*	FILE	Execute
/usr/sbin/user*	FILE	

Access Rights: Read, Write, [✓ Execute], Delete, Update, Full Control, Change Ownership, Change Permissions, No Access

② Highlight each resource and choose Execute access

③ Select **Assign Groups** to move to next dialog

Role Record Properties

Add Role Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Role Name: Acct Adm
Description:
Parent:
Action: Assign Groups

Group List

Selected Group List

Security Profiles: Examples-Profil, HQ_Systems-P
Available Groups: Account Admins
Assigned Groups: Examples-Profile:

④ Select source profile

⑤ Select group and add to list

⑥ Last thing, Create & Close

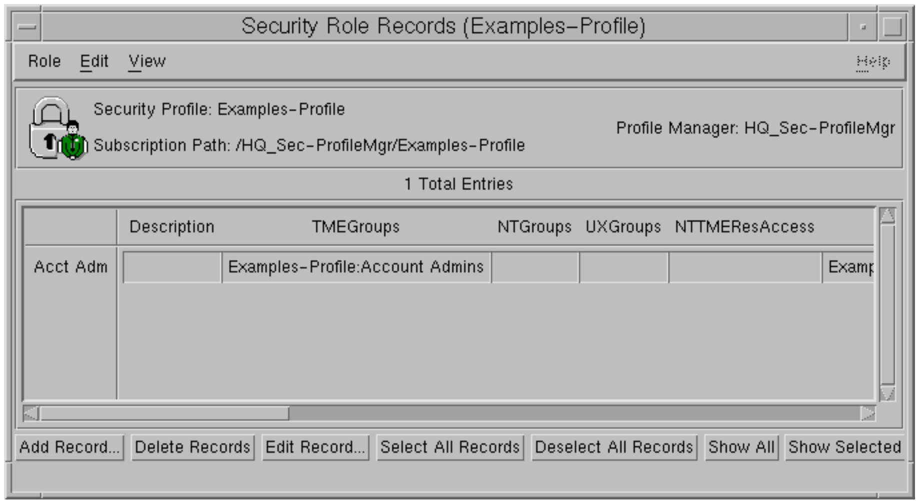
Figure 58. Selecting Resources and Assigning Groups to a Role Record

We now have an Acct Adm security role providing the requisite access to the Solaris user and group utilities for members of our Account Admins group.

The equivalent TME command to perform this same function looks like this:

```
wcrtsec Role -s TMEGroups='Account Admins' -s UXTMEResAccess= 'FILE:/usr/sbin/user*
  perms(X), FILE:/usr/sbin/group* perms(X), FILE:usr/bin/admintool perms(X)'
  @Example-Profile 'Acct Adm'
```

The role records dialog now lists the role we created (Figure 59).


Security Role Records (Examples−Profile)
Role Edit View
Help

Security Profile: Examples−Profile
Subscription Path: /HQ_Sec−ProfileMgr/Examples−Profile
Profile Manager: HQ_Sec−ProfileMgr

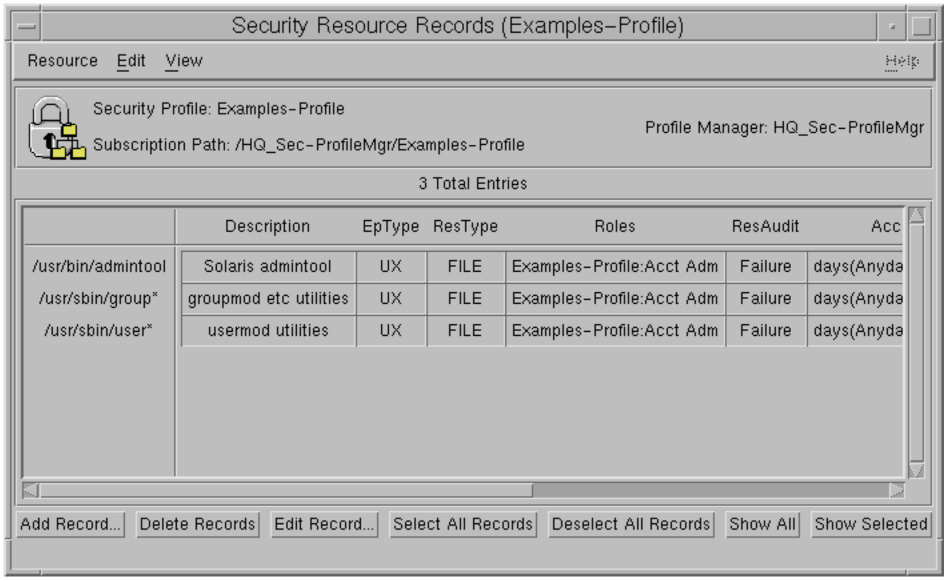
1 Total Entries

	Description	TMEGroups	NTGroups	UXGroups	NTTMEResAccess	
Acct Adm		Examples−Profile:Account Admins				Examp

Add Record... Delete Records Edit Record... Select All Records Deselect All Records Show All Show Selected

Figure 59. Completed Role Record

Note also that if we look again now at the resource record listing, the role attribute column is completed with the profile-qualified name of the role governing that resource (Figure 60).



Security Profile: Examples−Profile
Subscription Path: /HQ_Sec−ProfileMgr/Examples−Profile
Profile Manager: HQ_Sec−ProfileMgr

3 Total Entries

	Description	EpType	ResType	Roles	ResAudit	Acc
/usr/bin/admintool	Solaris admintool	UX	FILE	Examples−Profile:Acct Adm	Failure	days(Anyda
/usr/sbin/group*	groupmod etc utilities	UX	FILE	Examples−Profile:Acct Adm	Failure	days(Anyda
/usr/sbin/user*	usermod utilities	UX	FILE	Examples−Profile:Acct Adm	Failure	days(Anyda

Add Record... Delete Records Edit Record... Select All Records Deselect All Records Show All Show Selected

Figure 60. Resource Records Dialog with Roles Attribute Information

We are now ready to distribute this profile. In this environment, a user administrator would need to su to root and perform the user administration commands. This is because Solaris requires root to run the utilities. TACF allows these users to run the utilities as root because TACF checks the original (or effective) ID, not the ID they su'd to.

7.2.3 Defining Which Users Can su to root - Generic UNIX

Generally you are only going to want user IDs that are performing system administration to have the capability to su to root. With TME 10 Security Management this is achieved using the following steps:

1. Create a security resource that sets the default that no user can su to root.

   First, we select the **UNIX SURROGATE** resource type in a new resource record and edit default access to set it to None (Figure 61).

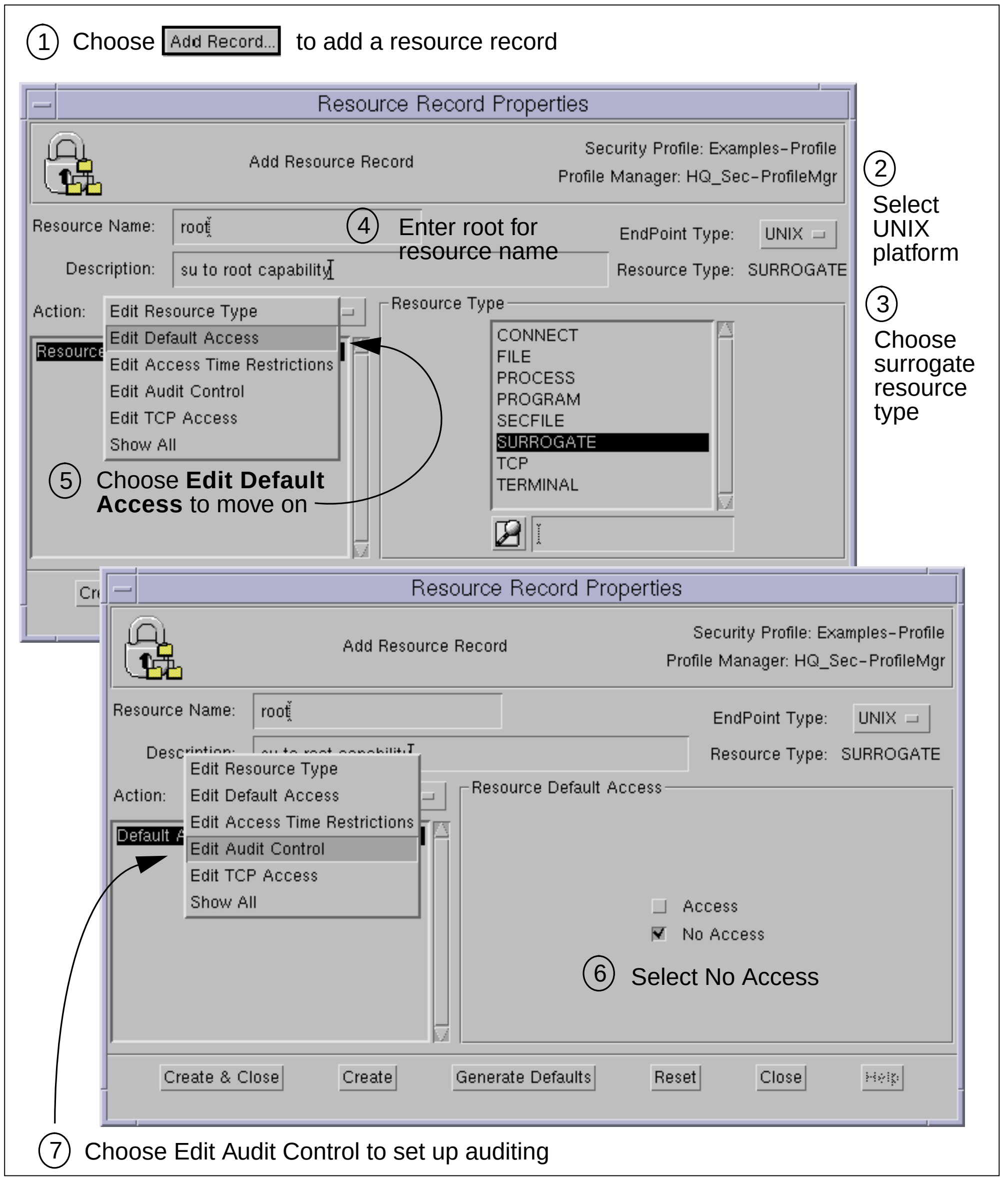
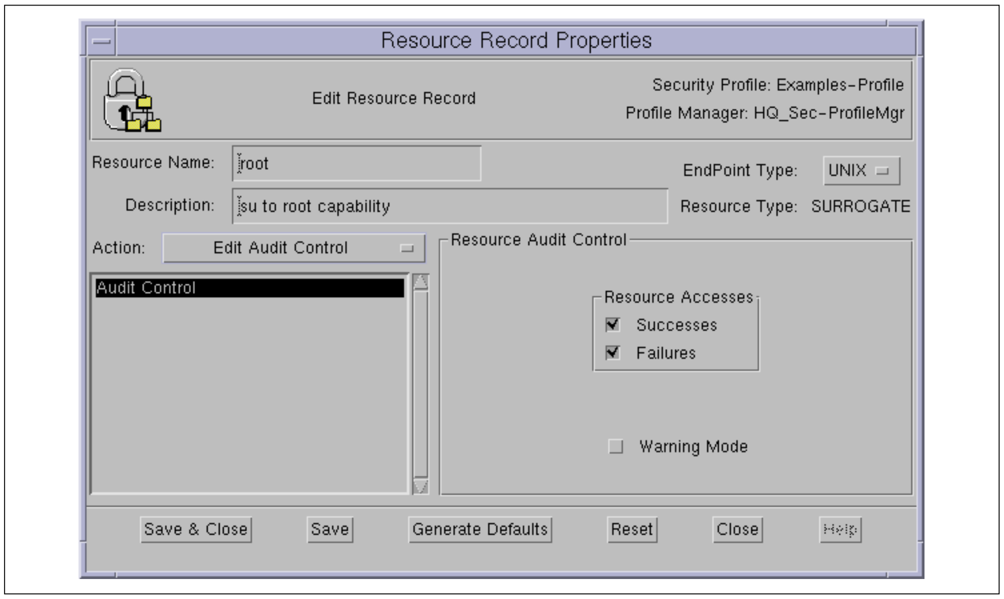


Figure 61. Defining a Surrogate Resource to Protect su to Root

Now we can set up to audit any attempts to su to root before we complete the creation of this record (Figure 62).


Resource Record Properties

Edit Resource Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Resource Name: root

Description: su to root capability

EndPoint Type: UNIX

Resource Type: SURROGATE

Action: Edit Audit Control

Audit Control

Resource Audit Control

Resource Accesses
- [✓] Successes
- [✓] Failures

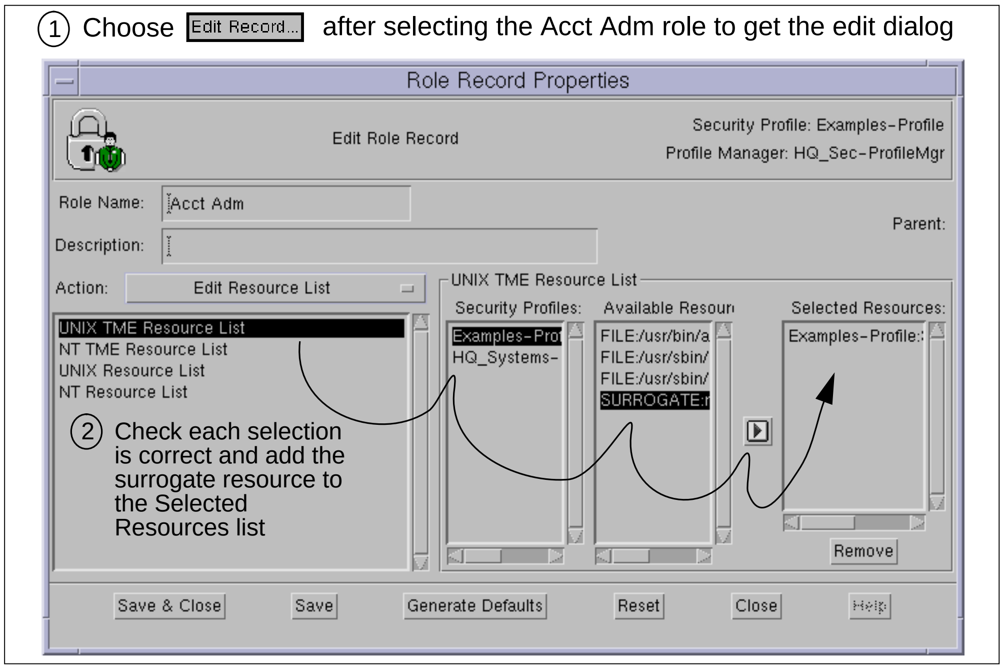
- [ ] Warning Mode

Figure 62. Setting Auditing on a Surrogate Resource

The equivalent TME command to perform this same function looks like this:

```
wcrtsec Resource -s DefAccess='perms(N)' -s ResAudit='All' @Example-Profile
  UX:SURROGATE:root
```

2. Now we modify a role to grant members of an administration group the ability to su to root (we use the Acct Adm role created in the previous example, giving access to the Account Admins group also created previously).


1 Choose Edit Record... after selecting the Acct Adm role to get the edit dialog

Role Record Properties

Edit Role Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Role Name: Acct Adm

Description:

Parent:

Action: Edit Resource List

UNIX TME Resource List
NT TME Resource List
UNIX Resource List
NT Resource List

UNIX TME Resource List

Security Profiles: Examples-Pro, HQ_Systems-

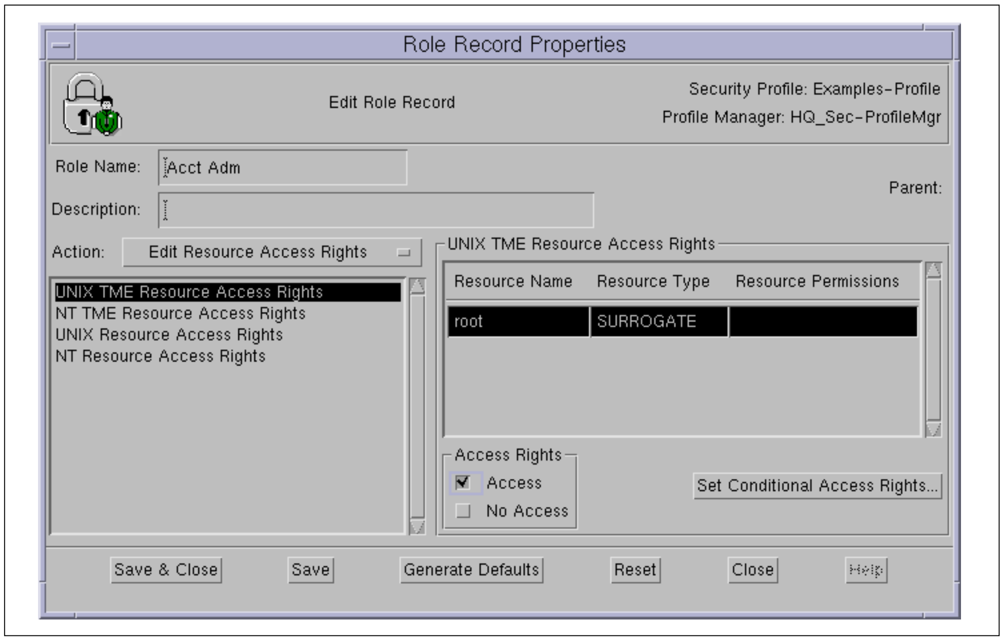
Available Resour: FILE:/usr/bin/a, FILE:/usr/sbin/, FILE:/usr/sbin/, SURROGATE:r

Selected Resources: Examples-Profile:

2 Check each selection is correct and add the surrogate resource to the Selected Resources list

Figure 63. Editing a Role Record and Adding a Surrogate Resource

Having put the surrogate resource on the resource list, we can then change the action to **Edit Resource Access Rights** and grant access to the surrogate resource to those assigned this role (Figure 64).


Role Record Properties

Edit Role Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Role Name: Acct Adm
Description:
Parent:
Action: Edit Resource Access Rights

UNIX TME Resource Access Rights
NT TME Resource Access Rights
UNIX Resource Access Rights
NT Resource Access Rights

UNIX TME Resource Access Rights

Resource Name	Resource Type	Resource Permissions
root	SURROGATE	

Access Rights
[✓ Access]
No Access

Figure 64. Editing UNIX TME Resource Access Rights

The equivalent TME command to perform this same function looks like this:

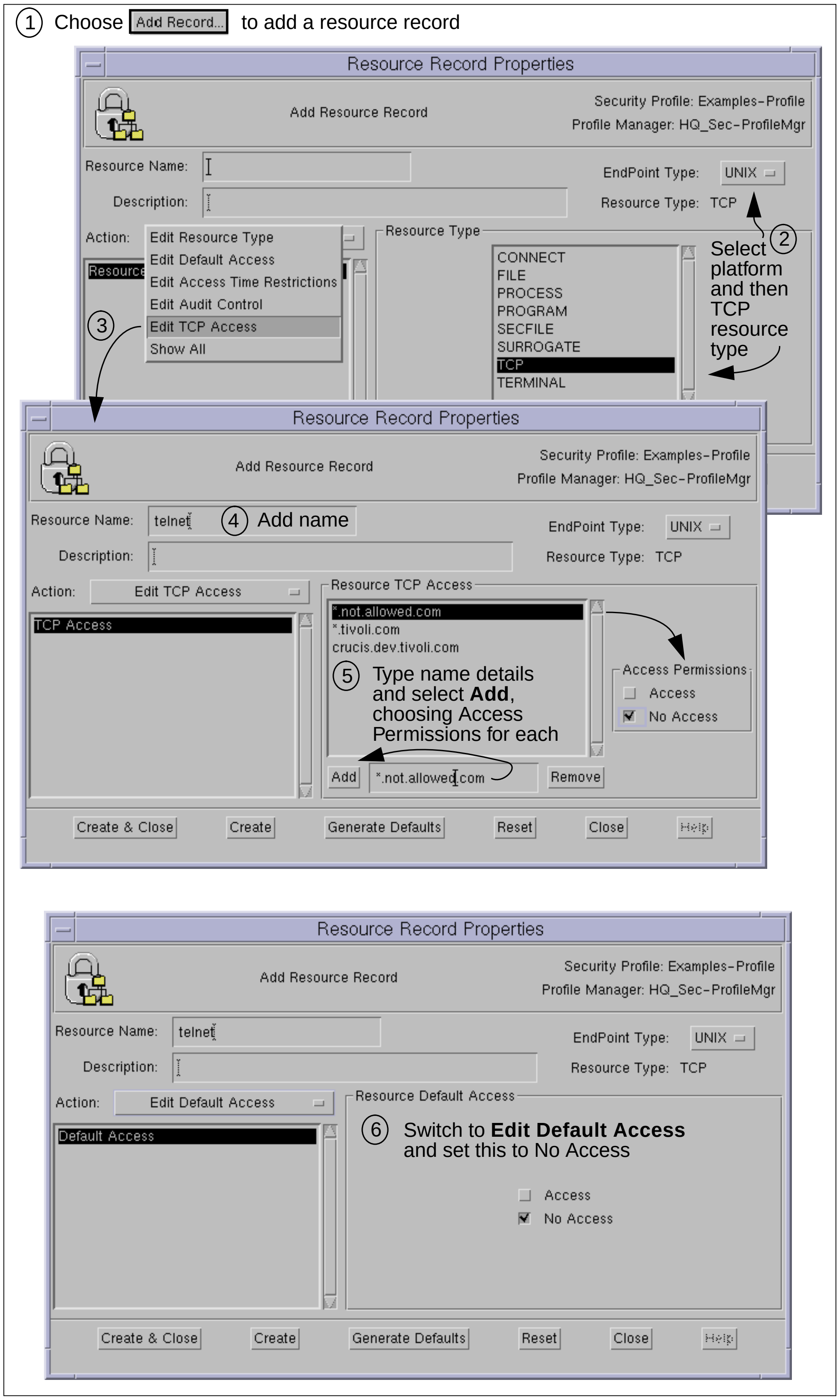
```
wmodsec Role -a UXTMEResAccess='SURROGATE:root perms(A)' @Example-Profile 'Acct Adm'
```

See Figure 67 to see the new resource and role assignment listed in the resource records dialog.

7.2.4 Restricting Incoming Network Requests - Generic UNIX

Now we will define a resource that will restrict the default access of a system through Telnet. We can specify many TCP/IP services such as FTP, rexec, rsh, and so on. We create a resource record that sets default access as allowed for a specific DNS name (crucis.dev.tivoli.com) and all the hostnames in a subnet (*.tivoli.com) as well as specifically denying access to one other subnet (*.not.allowed.com).

First, we create a new resource record for a resource of type TCP and then set the TCP Access.



(1) Choose Add Record... to add a resource record

Resource Record Properties

Add Resource Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Resource Name:
Description:
EndPoint Type: UNIX
Resource Type: TCP
Action:
Resource Type

(2) Select platform and then TCP resource type

(3)

Resource Record Properties

Add Resource Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Resource Name: telnet (4) Add name
Description:
EndPoint Type: UNIX
Resource Type: TCP
Action: Edit TCP Access

TCP Access

Resource TCP Access
*.not.allowed.com
*.tivoli.com
crucis.dev.tivoli.com

(5) Type name details and select **Add**, choosing Access Permissions for each

*.not.allowed.com

Access Permissions
Access
[✓ No Access]

Resource Record Properties

Add Resource Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

Resource Name: telnet
Description:
EndPoint Type: UNIX
Resource Type: TCP
Action: Edit Default Access

Default Access

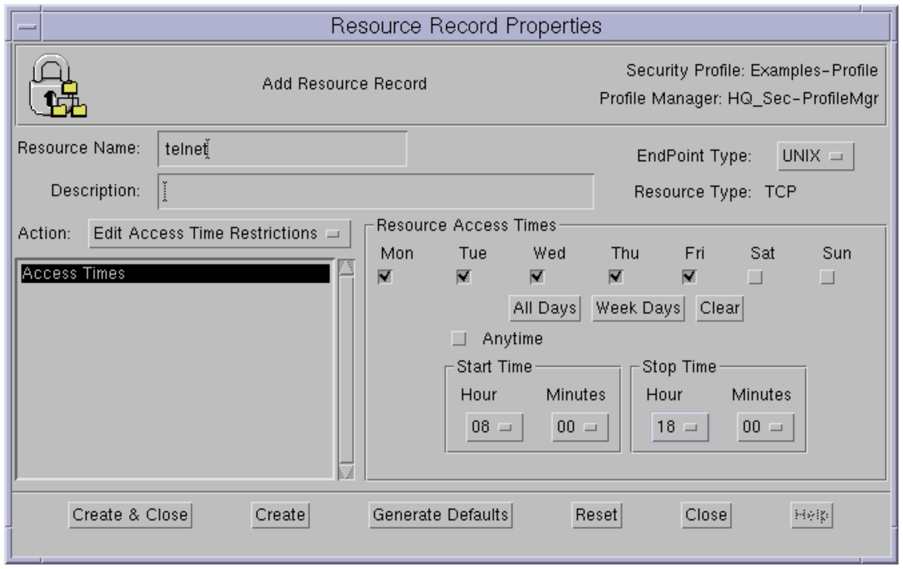
Resource Default Access

(6) Switch to **Edit Default Access** and set this to No Access

Access
[✓ No Access]

Figure 65. Define a Resource Record to Restrict Network Connection Requests

All that is left to do now is to add some time restrictions, and then we can create this resource (Figure 66). This is an example of creating access control by setting defaults for a resource. This could be further refined if required by giving specific groups roles with greater access.



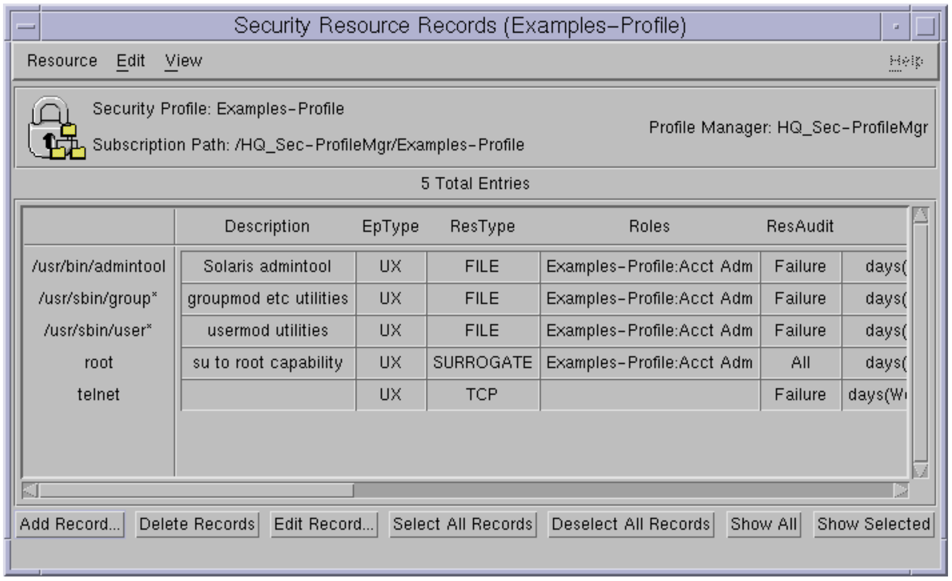
Security Profile: Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr
Resource Name: telnet
Description:
EndPoint Type: UNIX
Resource Type: TCP
Mon ✓ Tue ✓ Wed ✓ Thu ✓ Fri ✓ Sat ☐ Sun ☐
☐ Anytime
Start Time: Hour 08 Minutes 00
Stop Time: Hour 18 Minutes 00

Figure 66. Setting Access Time Restrictions for a TCP Resource

The equivalent TME command to perform this same function looks like this:

```
wcrtsec Resource -s TCPAccess='crucis.dev.tivoli.com perms(A), *.tivoli.com perms(A),
  *.not.allowed.com perms(N)' -s DefAccess='perms(N)' -s
  AccessTimes='days(Mon,Tue,Wed,Thu,Fri) time(0800:1800)' @Example-Profile UX:TCP:telnet
```

The resulting resource record is shown in Figure 67.



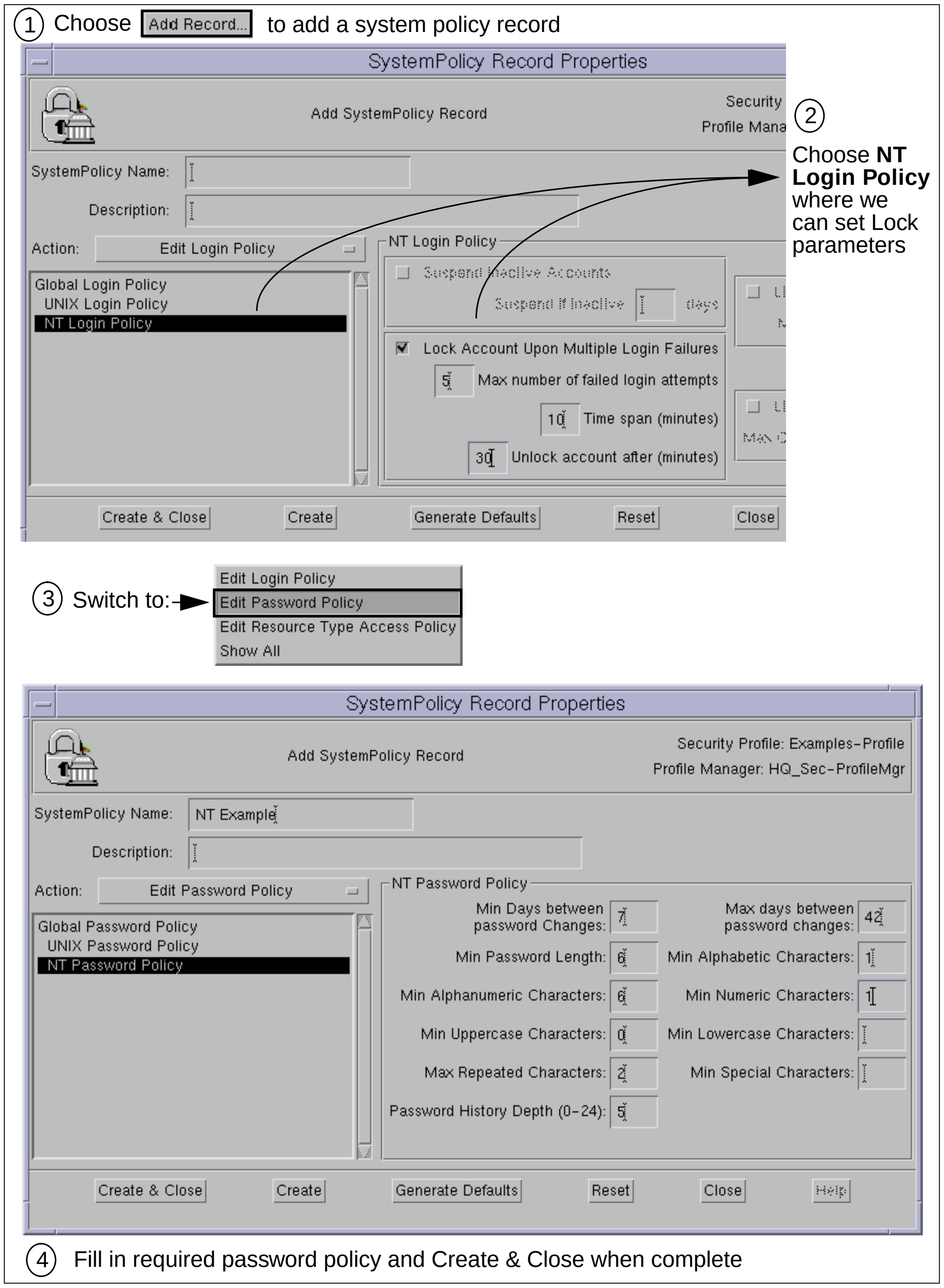
Security Profile: Examples–Profile
Subscription Path: /HQ_Sec–ProfileMgr/Examples–Profile
Profile Manager: HQ_Sec–ProfileMgr
5 Total Entries

	Description	EpType	ResType	Roles	ResAudit	
/usr/bin/admintool	Solaris admintool	UX	FILE	Examples–Profile:Acct Adm	Failure	days(
/usr/sbin/group*	groupmod etc utilities	UX	FILE	Examples–Profile:Acct Adm	Failure	days(
/usr/sbin/user*	usermod utilities	UX	FILE	Examples–Profile:Acct Adm	Failure	days(
root	su to root capability	UX	SURROGATE	Examples–Profile:Acct Adm	All	days(
telnet		UX	TCP		Failure	days(W

Figure 67. View of Completed Resource Records

7.2.5 Define Windows NT System Policy

This is a fairly simple example of how to set up a system policy for Windows NT. This is all performed through one system policy record defined in a security profile.



1 Choose Add Record... to add a system policy record

SystemPolicy Record Properties

Add SystemPolicy Record

Security Profile Mana

SystemPolicy Name:

Description:

Action: Edit Login Policy

Global Login Policy
UNIX Login Policy
NT Login Policy

NT Login Policy

☐ Suspend Inactive Accounts

Suspend if inactive days

☑ Lock Account Upon Multiple Login Failures

5 Max number of failed login attempts

10 Time span (minutes)

30 Unlock account after (minutes)

Create & Close | Create | Generate Defaults | Reset | Close

2 Choose **NT Login Policy** where we can set Lock parameters

3 Switch to:

Edit Login Policy
Edit Password Policy
Edit Resource Type Access Policy
Show All

SystemPolicy Record Properties

Add SystemPolicy Record

Security Profile: Examples-Profile
Profile Manager: HQ_Sec-ProfileMgr

SystemPolicy Name: NT Example

Description:

Action: Edit Password Policy

Global Password Policy
UNIX Password Policy
NT Password Policy

NT Password Policy

Min Days between password Changes:	7	Max days between password changes:	42
Min Password Length:	6	Min Alphabetic Characters:	1
Min Alphanumeric Characters:	6	Min Numeric Characters:	1
Min Uppercase Characters:	0	Min Lowercase Characters:	
Max Repeated Characters:	2	Min Special Characters:	
Password History Depth (0-24):	5		

Create & Close | Create | Generate Defaults | Reset | Close | Help

4 Fill in required password policy and Create & Close when complete

Figure 68. Defining a Windows NT System Policy

The equivalent TME command to perform the same function is:

```
wcrtsec SystemPolicy -s NTLockout='5,10,30' -s NTPwMinAge='7' -s NTPwMinLen='6'
  NTPwMinAlphanums='6' -s NTPwMinUppers='0' -s NTMaxReps='2' -s NTPwHistory='5' -s
  NTPwMaxAge='42' -s NTPwMinAlphas='1' -s NTPwMinNums='1'
```

Chapter 8. TME 10 Security Management Futures

This book was written around the first release of TME 10 Security Management. This release incorporates the proven technology of Memco's SeOS UNIX security product and provides two major benefits in networked systems management:

1. A sophisticated and consistent implementation of access control for disparate UNIX implementations
2. A single, integrated interface to managing complex UNIX- and NT-based networks

The TME 10 Security Management development team at Tivoli are working very hard to produce a number of major additions to the product that are following very closely behind the initial product release. Much of this work is already advanced at the time of writing; so we present some information on the work in progress.

Disclaimer

This text is provided for information purposes on features that may or may not be available through TME 10 Security Management in the future. Inclusion here of a description is no guarantee that these features will ever be made available in any or all countries in which TME 10 products are distributed.

8.1 Enhanced Scalability Release

The TME 10 Framework is already employed in many huge organizations and is managing environments both complex and simple. Tivoli has identified a number of factors that will ease the deployment and use of the major TME 10 applications across large institutions. This initiative is known as the Enhanced Scalability project, or ES1 (there are likely to be a number of waves of similar initiatives).

The aims of ES1 are to ensure major TME 10 applications, including TME 10 Security Manager, have the following capabilities:

- Support of the new Lightweight Client Framework (LCF) enabling a single to TMR to manage many thousands of end-nodes.
- Enable national language support (NLS) enabling products to be more easily translated and therefore shipped more quickly in major languages around the world.
- Integration of Application Management Specification (AMS) interfaces

See Appendix B, "Introduction to TME 10 Framework Version 3.2" on page 195, for a further discussion on the LCF.

8.2 Integration with MVS Security

TME 10 Security Management aims to provide the same integrated, centralized management of access control to MVS resources. This will be achieved through the ability to control MVS security products such as RACF. In anticipation of this capability, the following information is provided about the RACF environment.

Note that the MVS discussion is based around RACF. It may be that other access control products such as ACF/2 or Top Secret are employed to secure MVS systems, and at the time of writing, support for those products is intended but not finalized.

8.2.1 MVS

Perhaps surprisingly, native MVS contains less security than native UNIX. This is because MVS acknowledges that security is important enough to be handled by an external facility. That facility must provide and specialize in the needed granularity and scalability to provide flexible and robust security.

MVS identifies certain critical events in the OS as security-sensitive junctures. At these junctures, MVS uses an interface called the Security Authorization Facility (SAF) to call external security products. These external products maintain users' permissions and decide whether the requested actions are permissible. Based on the response it receives from the security engine, MVS either continues with the operation or return a denial message to the user.

The MVS approach provides all information relevant to making an authorization decision available to the external security engine, be it RACF, ACF/2 or Top Secret. This approach provides consistent, centralized security that controls events before they take place, maintains accurate accountability, produces a reliable and comprehensive audit trail, and lets extensions of the security system include specialized features.

User access permissions sometimes require that certain granular conditions be met. For example, a user's access to a database may require use of a particular application program, and login to the system may be limited to a time of day or day of week. For each decision, SAF lets MVS pass to the external security product all relevant information, such as user ID, the resource being accessed, the type of access requested (Read, Update, Alter), the access path, and so on. This information lets the external security engine support extensive Access Control List (ACL) structures and apply flexible and extensive conditions to the protection of resources

8.2.2 RACF Security Structures

IBM’s Resources Access Control Facility (RACF) licensed program provides a powerful product to secure an installation’s data. It really works closely with the operating system’s existing features, allowing the protection and control of the vital system resources. RACF will:

- Identify and verify system users
- Authorize the users who need access to the resources you've protected
- Control the means of access to these resources
- Log and report unauthorized attempts at gaining access to the system and to the protected resources
- Administer security to meet your installation's security goals.

RACF supports OS/390 by being a key component of the OS/390 Security Server. RACF groups all the resources it recognizes into *classes,* and every member of a specific class is called a *profile*. In other words, a profile is a member of a RACF class, and it represents a system resource.

But, before starting to define the resources to RACF (by adding profiles to classes), it is necessary to set up general RACF security options. The record that stores the general options is called *SETROPTS*.

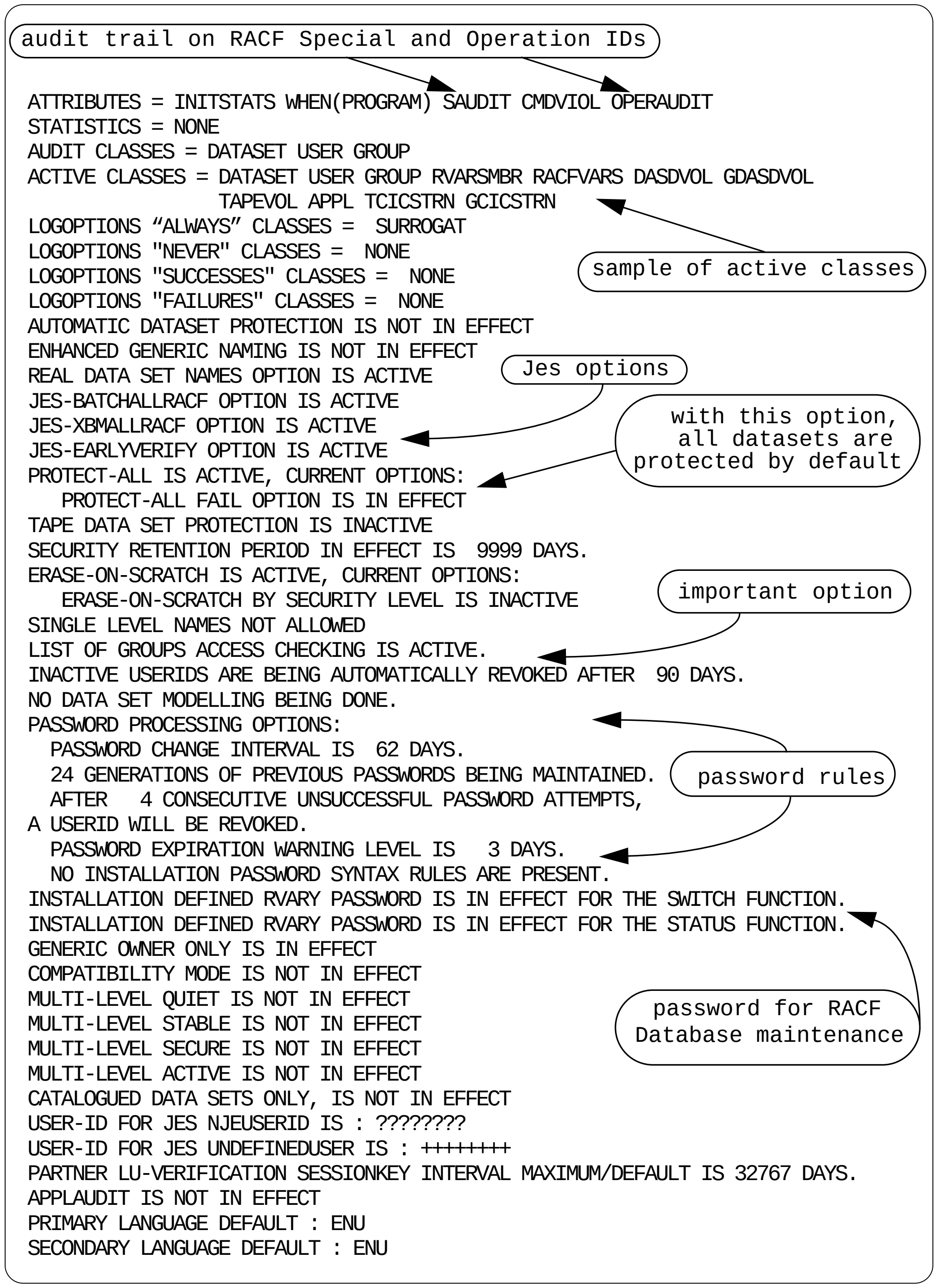
The following sections discuss the SETROPTS record and RACF classes. By default, RACF activates three basic classes, USER, GROUP and DATASET, and these are the ones discussed here. Other classes are known as General Resource classes.

8.2.2.1 The SETROPTS Record

In SETROPTS, RACF defines and stores the general rules of security and all its internal parameters:

- Default audit option
- The active classes
- Default dataset protection
- Job Entry Subsystem (JES) related information
- The password rules
- Rule to revoke inactive user IDs
- And many other options...

There are so many RACF options that a list of all of them would be huge. We show here the output of a `SETROPTS LIST` command, highlighting the basic options:

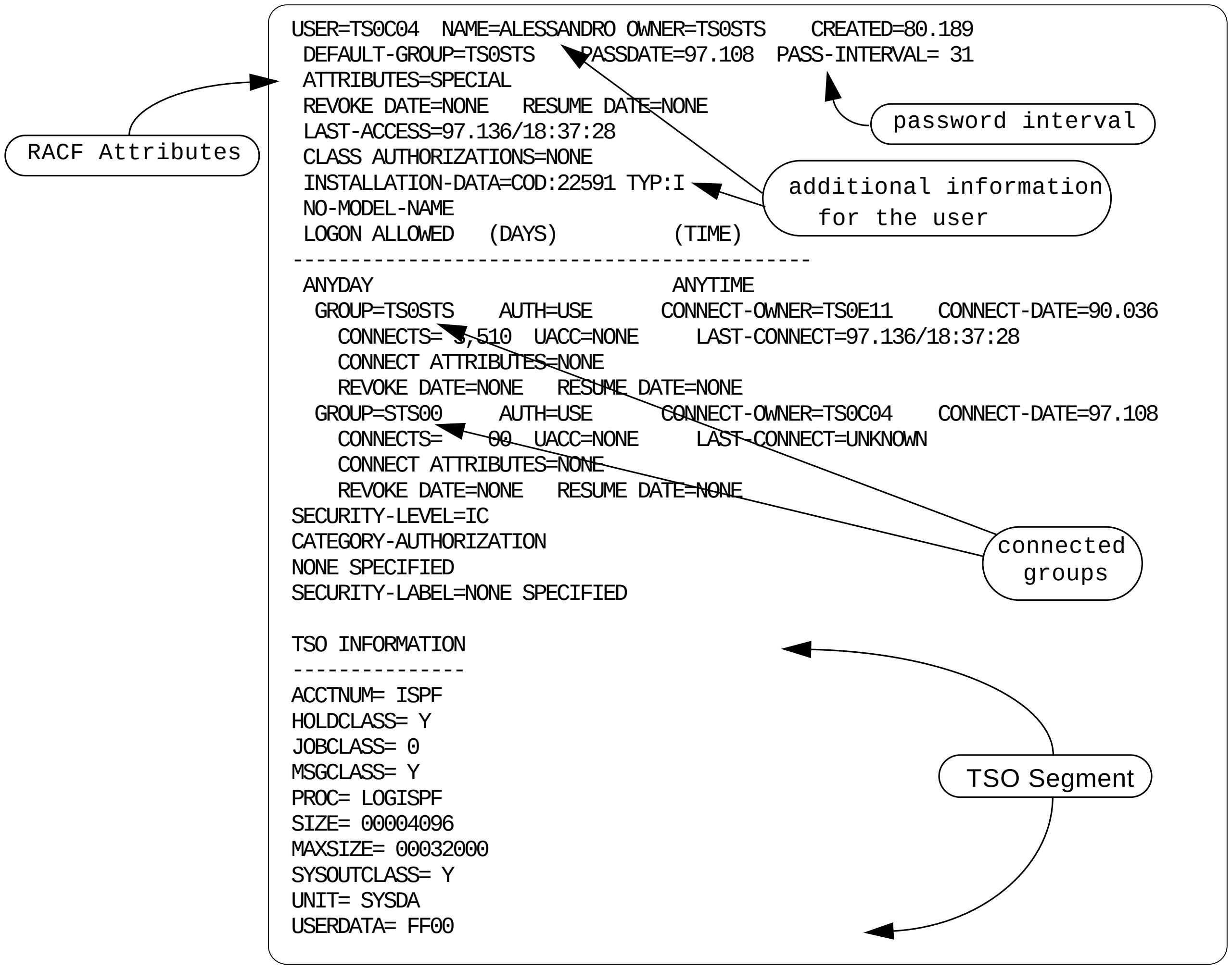

audit trail on RACF Special and Operation IDs
sample of active classes
Jes options
with this option, all datasets are protected by default
important option
password rules
password for RACF Database maintenance

```
ATTRIBUTES = INITSTATS WHEN(PROGRAM) SAUDIT CMDVIOL OPERAUDIT
STATISTICS = NONE
AUDIT CLASSES = DATASET USER GROUP
ACTIVE CLASSES = DATASET USER GROUP RVARSMBR RACFVARS DASDVOL GDASDVOL
                 TAPEVOL APPL TCICSTRN GCICSTRN
LOGOPTIONS “ALWAYS” CLASSES =  SURROGAT
LOGOPTIONS "NEVER" CLASSES =  NONE
LOGOPTIONS "SUCCESSES" CLASSES =  NONE
LOGOPTIONS "FAILURES" CLASSES =  NONE
AUTOMATIC DATASET PROTECTION IS NOT IN EFFECT
ENHANCED GENERIC NAMING IS NOT IN EFFECT
REAL DATA SET NAMES OPTION IS ACTIVE
JES-BATCHALLRACF OPTION IS ACTIVE
JES-XBMALLRACF OPTION IS ACTIVE
JES-EARLYVERIFY OPTION IS ACTIVE
PROTECT-ALL IS ACTIVE, CURRENT OPTIONS:
   PROTECT-ALL FAIL OPTION IS IN EFFECT
TAPE DATA SET PROTECTION IS INACTIVE
SECURITY RETENTION PERIOD IN EFFECT IS  9999 DAYS.
ERASE-ON-SCRATCH IS ACTIVE, CURRENT OPTIONS:
   ERASE-ON-SCRATCH BY SECURITY LEVEL IS INACTIVE
SINGLE LEVEL NAMES NOT ALLOWED
LIST OF GROUPS ACCESS CHECKING IS ACTIVE.
INACTIVE USERIDS ARE BEING AUTOMATICALLY REVOKED AFTER  90 DAYS.
NO DATA SET MODELLING BEING DONE.
PASSWORD PROCESSING OPTIONS:
  PASSWORD CHANGE INTERVAL IS  62 DAYS.
  24 GENERATIONS OF PREVIOUS PASSWORDS BEING MAINTAINED.
  AFTER   4 CONSECUTIVE UNSUCCESSFUL PASSWORD ATTEMPTS,
A USERID WILL BE REVOKED.
  PASSWORD EXPIRATION WARNING LEVEL IS   3 DAYS.
  NO INSTALLATION PASSWORD SYNTAX RULES ARE PRESENT.
INSTALLATION DEFINED RVARY PASSWORD IS IN EFFECT FOR THE SWITCH FUNCTION.
INSTALLATION DEFINED RVARY PASSWORD IS IN EFFECT FOR THE STATUS FUNCTION.
GENERIC OWNER ONLY IS IN EFFECT
COMPATIBILITY MODE IS NOT IN EFFECT
MULTI-LEVEL QUIET IS NOT IN EFFECT
MULTI-LEVEL STABLE IS NOT IN EFFECT
MULTI-LEVEL SECURE IS NOT IN EFFECT
MULTI-LEVEL ACTIVE IS NOT IN EFFECT
CATALOGUED DATA SETS ONLY, IS NOT IN EFFECT
USER-ID FOR JES NJEUSERID IS : ????????
USER-ID FOR JES UNDEFINEDUSER IS : ++++++++
PARTNER LU-VERIFICATION SESSIONKEY INTERVAL MAXIMUM/DEFAULT IS 32767 DAYS.
APPLAUDIT IS NOT IN EFFECT
PRIMARY LANGUAGE DEFAULT : ENU
SECONDARY LANGUAGE DEFAULT : ENU
```

8.2.2.2 The USER Class

In this class, it is possible to define the user IDs with their attributes. Every profile describes a user ID, and it is formed by the base segment plus additional product-specific segments. Segments relating to TSO, CICS, NetView and OMVS are the most security relevant.

As an example, the `listuser` command for the user ID TS0C04 produces the following output:



```
USER=TS0C04  NAME=ALESSANDRO OWNER=TS0STS    CREATED=80.189
 DEFAULT-GROUP=TS0STS    PASSDATE=97.108  PASS-INTERVAL= 31
 ATTRIBUTES=SPECIAL
 REVOKE DATE=NONE   RESUME DATE=NONE
 LAST-ACCESS=97.136/18:37:28
 CLASS AUTHORIZATIONS=NONE
 INSTALLATION-DATA=COD:22591 TYP:I
 NO-MODEL-NAME
 LOGON ALLOWED   (DAYS)          (TIME)
---------------------------------------------
 ANYDAY                          ANYTIME
  GROUP=TS0STS    AUTH=USE      CONNECT-OWNER=TS0E11    CONNECT-DATE=90.036
    CONNECTS=  3,510  UACC=NONE     LAST-CONNECT=97.136/18:37:28
    CONNECT ATTRIBUTES=NONE
    REVOKE DATE=NONE   RESUME DATE=NONE
  GROUP=STS00     AUTH=USE      CONNECT-OWNER=TS0C04    CONNECT-DATE=97.108
    CONNECTS=    00  UACC=NONE     LAST-CONNECT=UNKNOWN
    CONNECT ATTRIBUTES=NONE
    REVOKE DATE=NONE   RESUME DATE=NONE
SECURITY-LEVEL=IC
CATEGORY-AUTHORIZATION
NONE SPECIFIED
SECURITY-LABEL=NONE SPECIFIED

TSO INFORMATION
---------------
ACCTNUM= ISPF
HOLDCLASS= Y
JOBCLASS= 0
MSGCLASS= Y
PROC= LOGISPF
SIZE= 00004096
MAXSIZE= 00032000
SYSOUTCLASS= Y
UNIT= SYSDA
USERDATA= FF00
```

In the example, the user ID TS0C04 has the RACF `Special` attribute. The complete list of attributes and their meaning as follows:

SPECIAL	System SPECIAL authority is the highest authority for RACF administration. It can add, change, remove any RACF definition.
OPERATION	System OPERATION is a very powerful RACF authority: it has almost every authority on DATASET and DASDVOL resources. But, be careful. Since RACF has a resources-oriented approach, the access list prevails on this attribute. This means RACF first makes a check on the access list, then on the userID attribute. If in the access list it is specified the user ID can only read, then it cannot update, even if it has the OPERATION attribute.
AUDITOR	By the assignment of this authority, a user ID can practice RACF auditors' activities. Basically, this authority allows a user to browse all the RACF definitions, to run Auditing utilities such as DSMON, and to modify Audit parameters.
REVOKE	This attribute means the user ID is locked.
NONE	It is the default attribute.

8.2.2.3 The GROUP Class

The RACF group can assume a lot of meanings. In fact, it can be:

- A user ID's default group
- A group of user IDs with the same access authorities
- An owning group
- A dataset prefix

The first and second definition are quite similar; the difference is that RACF requires to connect a user ID to a group during its definition. This means that a user ID can be connected to a lot of groups, but one connection is significant, since this group is also the default group. From an administration point of view, it is advisable to leave this group just as a default group, then connect the user ID to more meaningful groups. In this way, the structure is easier to understand and to manage, since to disconnect a user ID from a default group requires some particular actions.

Using the third definition, some RACF administrative actions are allowed. If you look at the List of TS0C04 user ID, you can see the owner is the TS0STS group (by chance, it's also the Default group). If a user ID is connected to the TS0STS group with the Special connect attribute, it can administer (for instance, reset the password) the TS0C04 user ID.

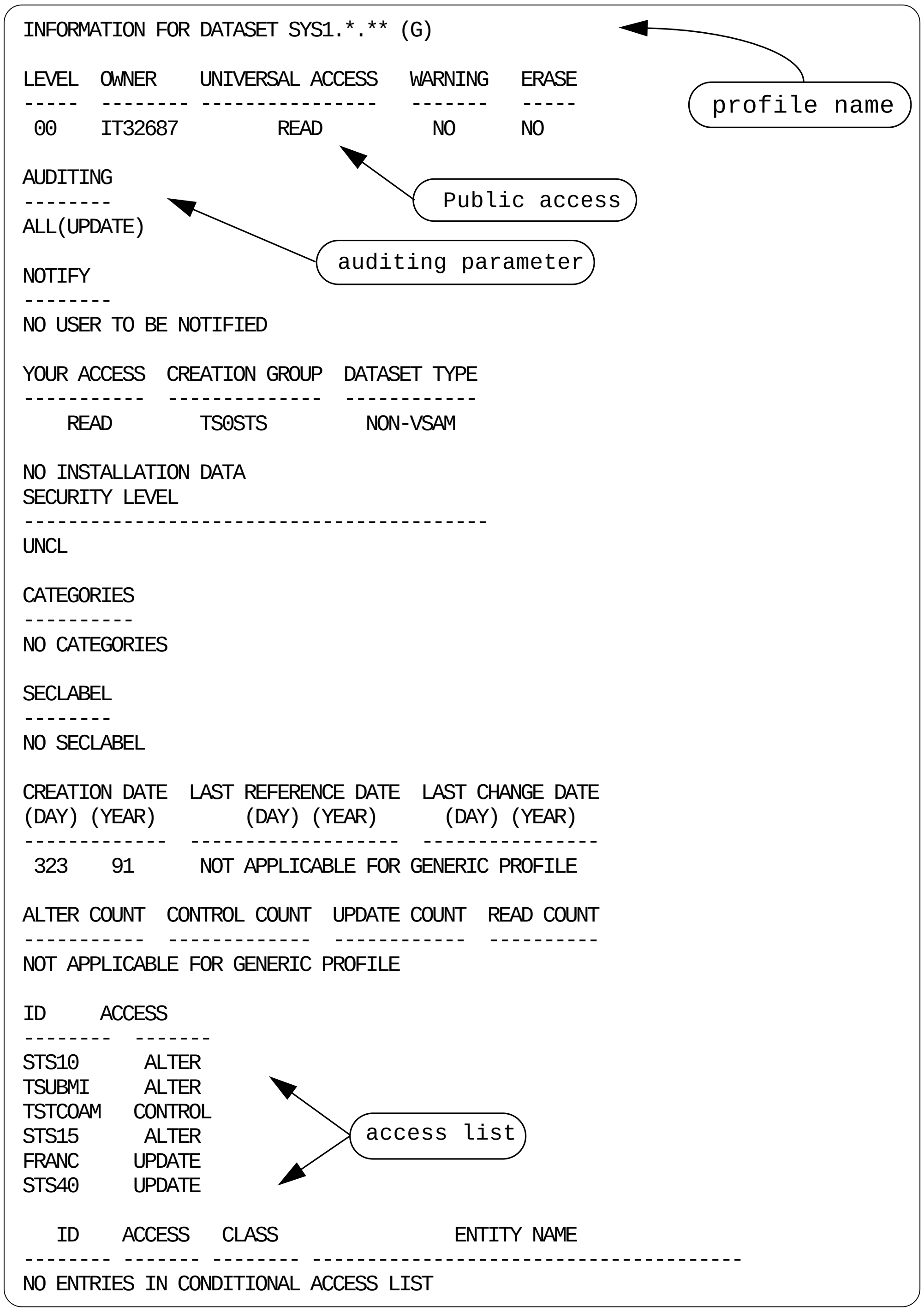
The fourth definition is completely different. The group is the prefix of a dataset. Nevertheless, by connecting a user ID to the group with particular attributes (for example, CREATE), the administration of DATASET is allowed.

Here is the output of a `listgroup` command for the Group STS00:

```
INFORMATION FOR GROUP STS00
    SUPERIOR GROUP=TS0STS       OWNER=TS0STS
    INSTALLATION DATA=PRJ:XSM
    NO MODEL DATA SET
    TERMUACC
    NO SUBGROUPS
    USER(S)=      ACCESS=      ACCESS COUNT=      UNIVERSAL ACCESS=
TS0C04        USE          000000             NONE
         CONNECT ATTRIBUTES=NONE
         REVOKE DATE=NONE                 RESUME DATE=NONE
IT25944       USE          000000             NONE
         CONNECT ATTRIBUTES=NONE
         REVOKE DATE=NONE                 RESUME DATE=NONE
IT09486       USE          000000             NONE
         CONNECT ATTRIBUTES=NONE
         REVOKE DATE=NONE                 RESUME DATE=NONE
```

8.2.2.4 The DATASET Class

In this class, the MVS datasets are described. The profile name can be generic (for example, SYS1.*.**) or discrete (for example, SYS1.PARMLIB). The access list consists of user IDs and groups and their related authorities: Read, Update, Control, Alter, Execute, or None, as described below.



```
INFORMATION FOR DATASET SYS1.*.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    IT32687        READ           NO      NO

AUDITING
--------
ALL(UPDATE)

NOTIFY
--------
NO USER TO BE NOTIFIED

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    READ        TS0STS         NON-VSAM

NO INSTALLATION DATA
SECURITY LEVEL
-------------------------------------------
UNCL

CATEGORIES
----------
NO CATEGORIES

SECLABEL
--------
NO SECLABEL

CREATION DATE  LAST REFERENCE DATE  LAST CHANGE DATE
(DAY) (YEAR)       (DAY) (YEAR)      (DAY) (YEAR)
-------------  -------------------  ----------------
 323    91      NOT APPLICABLE FOR GENERIC PROFILE

ALTER COUNT  CONTROL COUNT  UPDATE COUNT  READ COUNT
-----------  -------------  ------------  ----------
NOT APPLICABLE FOR GENERIC PROFILE

ID      ACCESS
--------  -------
STS10      ALTER
TSUBMI     ALTER
TSTCOAM   CONTROL
STS15      ALTER
FRANC     UPDATE
STS40     UPDATE

   ID    ACCESS   CLASS                ENTITY NAME
-------- ------- -------- ---------------------------------------
NO ENTRIES IN CONDITIONAL ACCESS LIST
```

With the generic profile definition, it is possible to protect a large collection of datasets with a few definitions. In the example, the SYS1.*.** is sufficient to protect all the datasets with the prefix SYS1. Additional dataset profiles could be added to differentiate the access rights

- SYS1.ABC.**

  It can cover the following datasets:

  - SYS1.ABC

    - SYS1.ABC.A12
    - SYS1.ABC.A12.BER
- SYS1.*.ABC

  It can cover the datasets

    - SYS1.XYZ.ABC
    - SYS1.WW.ABC

It is important to understand that a profile like SYS1.ABC.** prevails on the more generic SYS1.*.**. If a user needs to update a SYS1.ABC.A12 dataset, he/she needs to access the first profile even if he/she has ALTER access on SYS1.*.**.

The Access Parameter

As we have seen before, it is possible to specify different access rights:

ALTER	Allowed to delete or to allocate a dataset.
UPDATE	Allowed to update a dataset.
READ	Allowed to read a dataset.
EXECUTE	Allowed to execute the programs contained in the specified dataset: it doesn’t imply the ability to copy the program.
NONE	The access is denied.

The access rights are hierarchical, they follow the specified sequence.

8.2.2.5 The General Resources Class

General Resources consists of a seemingly endless list of RACF classes; every class is related to a specific kind of MVS resource. The most common classes are TAPEVOL, DASDVOL, PROGRAM, TERMINAL, CICS transactions, or SURROGAT, but many others are defined or can be customized. In every class, generic profiles can be defined; the basic information is very similar to the DATASET class, but other specific information can be defined.

8.2.3 Critical MVS Resources

Critical MVS operating system resources are discussed in this section. It is assumed here that RACF is the reference access control system: it allows us to show directly the protection to set up.

8.2.3.1 General Rule

Generally speaking, operating system resources are not read sensitive; the general rule is that every user can *read* system resources.

However, every rule can have its exceptions, and here we have ours. In fact, system resources must exempt the general rule when:

- They contain passwords
- They contain classified information (system dumps can be in this category)
- They provide detailed information about the system, and this information could be useful for penetration attempts

8.2.3.2 Operating System Resources Exceptions

At the time an MVS subsystem is installed, the general rule of granting read permission to everybody should be reevaluated for the following items:

Data Sets

- Page data sets
- Spool data sets
- Trace data sets
- System dump data sets
- SMF data sets
- RACF data bases (Primary, Secondary, Back-up versions)
- TSO - UADS
- JES3 - INISHDECK (RJP RJE)
- All system and subsystem PARMLIBs
- DFHSM High-Level Qualifier for backup and migration data sets
- Print Queue when it contains data (for example, ADMPRINT supplied by GDDM)
- Data set allocated to NPM DD name FNMPARM (if RACF is not used for logon security)
- NETVIEW/AS file EMSALLT (logon recording of users)
- NFT Parmlib (DD SYSIN) (misuse of queue handler, Queue master PW)
- NetView DSIPARM (if not using RACF as security manager)
- NetView Trace data sets
- NetView data set with passwords for automated operations, provided with the product asCNM.AOC.AOFPSWD
- NDM - Needs user ID and password during generation
- NDM - DSXDRD contains user IDs and passwords
- IBTS masterkey and backup masterkey data sets
- VTAM - VTAMLST (protect against switched major nodes)
- RODM - installation datasets (need a user ID/password) for generation

General Resources

It must be ensured that all the online volumes are protected through the RACF DASDVOL class. Otherwise, the following PROGRAMs must be controlled:

- ADRDSSU
- IGWSPZAP/AMASPZAP/IMASPZAP
- ICKDSF
- IDCSC01
- IEHATLAS

The following programs must be ALWAYS controlled:

- IEHINITT
- PX021062
- CREDISPO

Other resources to control are:

- ICHBLP (bypass Standard Label tape processing), using the RACF class FACILITY
- In RACF class SDSF, the ISFOPER.SYSTEM profile
- In RACF class TSOAUTH, ACCT, CONSOLE, OPER, PARMLIB, and TESTAUTH profiles
- In RACF class OPERCMDS, JESx.VS* profile

8.2.3.3 Critical RACF Administrative Authority

The Security Administrative Authorities are a kind of resource very critical for the system, too. This is the reason they should be always kept under control and the Business Need of the assigned users should be verified periodically.

In RACF, the following authorities could be identified as Security Administrative Authority:

- System SPECIAL user attribute

  System SPECIAL authority is the highest authority for RACF administration. It can add, change, and remove every RACF definition, but, by default, it cannot access any resources. Of course, it can permit itself, but this action must follow a Business Need; otherwise it can be considered a misuse of authority.

- System OPERATION user attribute

  System OPERATION is a very powerful RACF authority; it has almost every authority on DATASET and DASDVOL resources. In this sense, a System Operation user ID should be classified as System Administrative Authority, but, due to its significant authority, has been included in the Critical Authority list.

- System AUDITOR user attribute

  By the assignment of this authority, a user ID can practice RACF auditor activities. Basically, this authority allows a user to browse all the RACF definitions, to run Auditing utilities, such as DSMON, and to modify audit parameters.

- Group Connect Attribute of SPECIAL and the Class Authority (CLAUTH) of USER
- Group JOIN Authority and the Class Authority (CLAUTH) of USER

  Class Authority of USER, associated with Group SPECIAL attribute or Group JOIN Authority, allows the administration of user IDs (AddUser, AlterUser, DeleteUser) belonging to the scope of the Group.

8.2.4 RACF TME 10 Endpoint Support

Due to the peculiarity of MVS operating system, some explanations are needed to understand how the TME 10 environment can be connected to the RACF/MVS system. This discussion looks at the OS/390 Connection Service and current and soon-to-be released facilities for communication between the TME 10 environment and MVS or RACF. The direction for MVS support includes getting the TME 10 Framework on MVS and applications including TME 10 Security Management are to be enhanced to work with an MVS-based TME 10 Framework.

8.2.4.1 What is the Global Enterprise Manager?

The Global Enterprise Manager (GEM) product is intended to be a bridge between the OS/390 host environment and the distributed environment. It is built to facilitate systems and network management. The distributed environment could consist of UNIX, Windows NT, Novell, OS/2, and other platforms.

There are several separately installable entities within the OS/390 side of GEM, the OS/390 Connection Service being one of them.

The OS/390 Connection Service administers security tasks across multiple platforms with a single action. It manages System Authorization Facility (SAF) compliant products such as RACF and CA ACF/2. SAF is the component in OS/390 which responds to standard SAF service calls. SAF routes these calls to any security manager that is coded to understand these SAF-compliant calls. By using the OS/390 Connection Service, you can increase security through faster updates and consistent administration of policy while eliminating the need for administrators to log on to different systems.

The OS/390 Connection Service is the OS/390 component that responds to requests originating from the TME 10 oserv daemon running on the TMR server or on a TME Management Station. It is implemented as an OS/390 address space and contains the TCP/IP protocol support to manage the communications link between the Transaction Processing Server (TP server) and TME 10.

In order to ensure that requests coming from the TMR server are from authorized users, the OS/390 Connection Service uses link encryption and authentication of incoming requests. Standard SAF services are used to process the requests.

Since the OS/390 Connection Service is supposed to be a bridge between the host environment and the distributed environment, there are products corresponding to it on the TME 10 side. These products are in addition to the TME 10 Framework:

- TME 10 GEM OS/390 Connection Service
- TME 10 GEM Security Management Service for OS/390 (Tentative product name)

The TME 10 GEM OS/390 Connection Service enables you to send commands using TCP/IP communication to the OS/390 Connection Service (also called the TP server) on the host side. The TME 10 GEM Security Management Service is used to manage security-related information in a distributed environment, including the host environment with RACF or equivalent products.

8.2.4.2 Main Components and General Flow

The RACF/MVS TME 10 Security Management components and their relevant corequisites can be summarized in the following list:

- TME 10 GEM: OS/390 Connection Service - TP Server

  The TP server should be setup as a Started Task to provide Connection Services between the TMR Server and the MVS endpoint.

  NOTE

  The Lightweight Client Framework will have many of the capabilities of the full Tivoli Management Framework (TMF), but it will lack a local database. Once the LCF is available for MVS systems, we should no longer need the TP server infrastructure.

- TME 10 GEM: Security Management Services (Tentative product name)

  The extension for TME 10 to invoke MVS specific method

- RACF/SAF enhancement: the R_admin callable service

- Transmission Control Protocol/Internet Protocol (TCP/IP) for MVS, Version 3 Release 1 or later

Figure 69 on page 174 shows a high-level flow; it explains how the single components interact with each other.

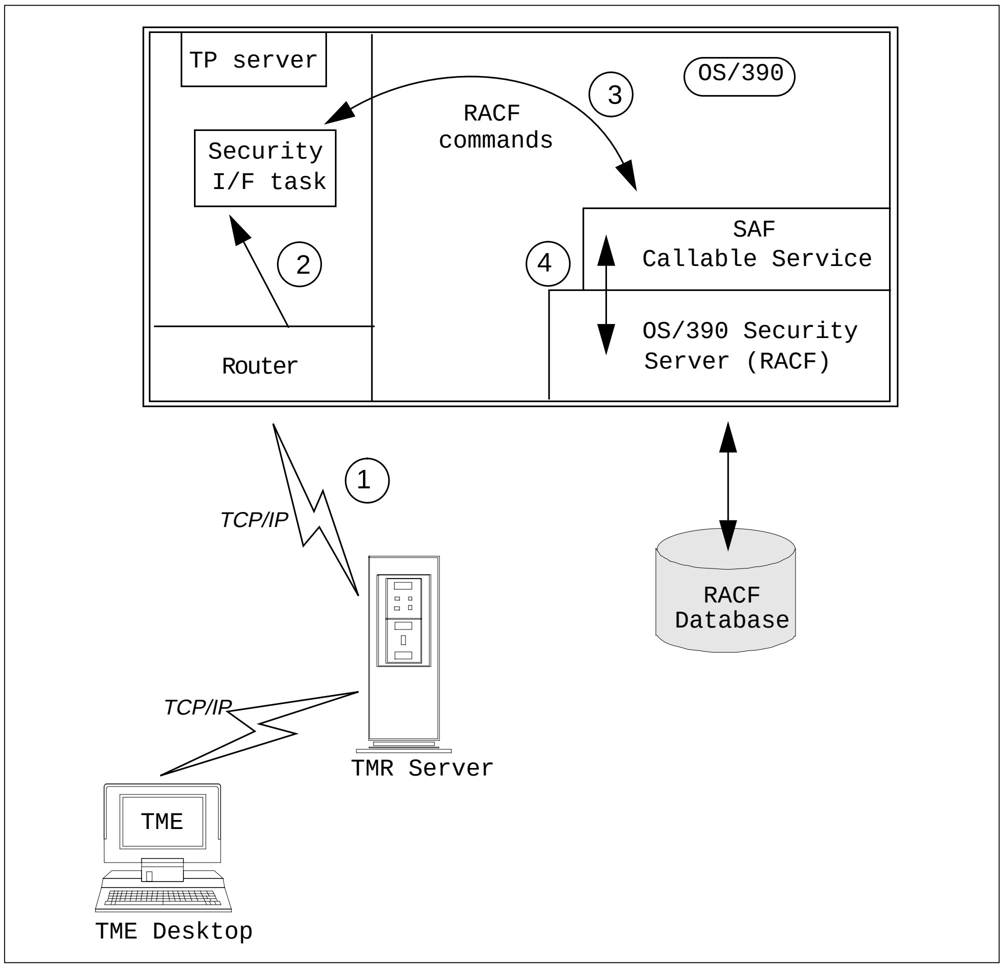


Figure 69. TMR to RACF High-Level Flow

Basically, these are the main steps:

1. The OS/390 TP server router receives packets from the TMR server through TCP/IP.
2. The OS/390 TP server router invokes the security I/F task.
3. The security I/F task invokes a new callable service, IRRSEQ00.
4. The SAF callable service, IRRSEQ00. in turn invokes IRRREQ00 (a RACF service) to carry out the security request.

8.2.4.3 OS/390 Security Server (RACF) PassTicket Technology

A secure communication between the TMR server and the OS/390 endpoint system is required in order to ensure that another application cannot act as the TMR server. In such a case, the risk is an unauthorized intrusion into the MVS system and the possible alteration of security data stored in the RACF data base.

In order to provide a secure communication, it is advisable to use a security session mechanism such as PassTicket. PassTicket is a RACF feature. Using this technology, the TMR server authenticates itself to the managed OS/390 node, avoiding the use of a RACF password.

The PassTickets are one-time use password substitutes and have a life span limited to ten minutes after their generation. After that, they are no longer valid as authenticators.

The PassTicket Algorithm requires a secret application key to be shared between RACF and the application which generates the PassTicket. The secret keys are stored in the RACF database on OS/390 systems and in the TME profile database on the TMR server. They never flows across the wire.

As shown in Figure 70 on page 175, the inputs to the PassTicket generator are the user ID of the TME administrator, the APPLID, the GMT time, and the secret key.

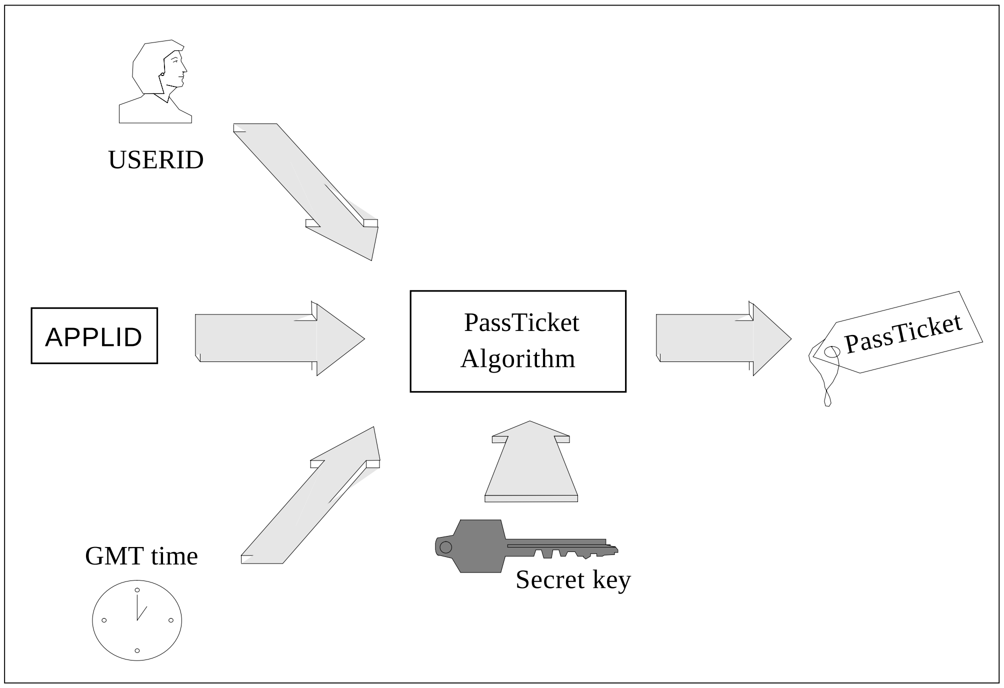


Figure 70. RACF PassTicket Generation

During the initial bind of the TMR server to OS/390, the TMR server generates the PassTicket and sends it with the TME administrator user ID to the MVS endpoint.

In Figure 71 on page 176, the flow is shown.

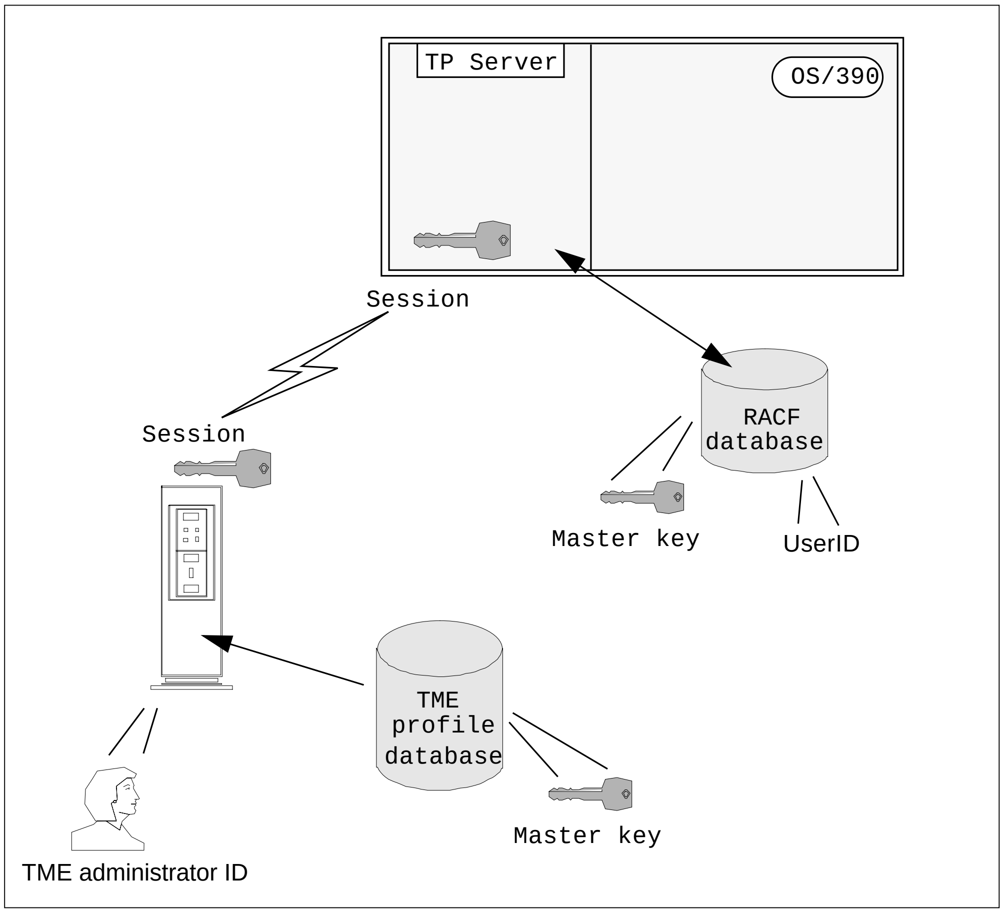


Figure 71. TMR and RACF PassTicket Flow

The OS/390 endpoint receives the request to start the communication with the TME 10 application. First, RACF checks if the TME administrator ID has a valid password. Once RACF recognizes that the TME administrator ID is not using a password to authenticate itself, it checks if the string of data received from the TMR server is a valid PassTicket. RACF generates in turn the session key, that is the PassTicket and verifies the TME administration ID authenticity. If the check is positive, the communication can start.

8.3 OS/2 Warp Server and Novell NetWare Support

OS/2 Warp Server will soon be supported as a full-fledged managed node (as opposed to the PC-managed node currently implemented), and with the availability of DCE management through the TME 10 Plus module from Santix, OS/2 Warp servers in a heterogeneous environment could come under Tivoli TME 10 management using the DCE-based IBM OS/2 Directory and Security Server (DSS). (The Santix DCE product will provide the necessary DCE extended registry attributes support which would enable full OS/2 DSS control by the end of 1997).

8.3.1 OS/2 Warp Server Security Structures

OS/2 Warp Server and associated products such as the Workspace-on-Demand Manager, implement much the same security features on HPFS-based servers as

Windows NT does on NTFS. Users are authenticated at the domain level, allowing access to resources controlled by ACLs. ACLs are maintained within the file system (in the case of HPFS) or in the NET.ACC file of the server (in the case of FAT or other non-HPFS resources such as printers). Access control is managed by collecting users in groups and providing either groups or individual users access rights over resources.

OS/2 Warp Server does not implement the trusted domain models for multidomain management. Instead, IBM uses the DCE-based Directory and Security Server (DSS) product to scale to large enterprises, providing DCE Cell naming and security capabilities. Installations using DSS should investigate the use of TME 10 Partner products such as the santix DCE Security Manager.

8.3.2 Critical OS/2 Warp Server Resources

OS/2 Warp server protects its own resources when local security is enabled on HPFS partitions. The items that need to be protected are:

- CONFIG.SYS
- STARTUP.CMD
- PRIVINIT.CMD
- \OS2 directory
- \IBMLAN directory - in particular the DCDB subdirectory, IBMLAN.INI and the NET.ACC file

8.3.3 Novell NetWare Security Structures

In this section, we discuss NetWare Versions 3.x and 4.x as two different subjects. The security structure on them is quite similar, but they have some important differences we must take into consideration.

8.3.3.1 Novell NetWare Version 3.x

NetWare Version 3.x is managed with bindery-based security. Administering the security of NetWare 3.x is not too different from administering earlier versions. The names of the access rights changed from Version 2.15, and later versions added the supervisor access rights.

NetWare 3.x versions store security in three bindery files located in the SYS:SYSTEM directory. These bindery files are hidden and Read-Only protected. These three bindery files are:

NET$OBJ.SYS (objects)

NET$PROP.SYS (properties)

NET$BVAL.SYS (properties data sets)

A bindery file is a network database that contains definitions for entities, such as users, groups and workgroups.

Understanding How NetWare 3.x Bindery Works

When user GEORGE logs in to a file server, the login program looks in the NET$OBJ.SYS for the object name to determine if this is a valid user.

If an object named GEORGE exists, the program looks in the NET$PROP.SYS file for the properties associated with that object (in this case, to see if a password property for user GEORGE exists).

If GEORGE has a password property, the program prompts him for his password and compares this value with the value in the net$val.sys file that is assigned to the password property.

If the two values match, GEORGE is logged in and allowed to use that network's resources according to the values of other properties (such as account restrictions and trustee assignments) that exist for user GEORGE.

Password Encryption

Passwords are stored in encrypted form in the net$val.sys bindery file and are sent to the server encrypted using a hashing function. The three password functions *(login, change password, verify password)* have a secure protocol, meaning that the information gathered by packet sniffers cannot be used to reconstruct the event or determine the password (packet sniffers are programs that can capture information that is transmitted on the LAN, such as Telnet/FTP passwords, and so forth).

Note

Always disable the use of unencrypted passwords by running on the server the command `set allowed unencrypted passwords off`, or add it to the autoexec.ncf file.

NetWare 3.x Security

NetWare 3.x security consists of a combination of the following:

- Login security includes creating user names and passwords and imposing station, time and account restrictions on users.
- Trustee rights (privileges) assigned to users.
- Attributes assigned to directories and files, which determine whether the directory or file can be deleted, copied, viewed, or written to. Among other things, they also mark a file as shareable or non-shareable.

What are Trustees, Groups and Trustees' Rights?

A trustee is any user or group that has been granted access rights in a directory.

Groups are an object type which allows users to be banded together for different purposes. The main interest in this concept is to ease maintaining security, by granting access rights to groups instead of individual users. By default, all users are in a group called EVERYONE. If you grant rights to group EVERYONE for a directory, then every user in that group holds those rights over that directory.

Trustee rights control access to which directories and files a user can have access to and what the user is allowed to do with those directories and files, such as creating, erasing, or writing to them. The following is a summary of trustee access rights for NetWare 3.x:

S - Supervisor	Any user with supervisor rights in a directory automatically inherits all other rights, regardless of whether they have been explicitly granted or not. Supervisor equivalent accounts holds this access right in every directory.
R - Read	Allowed to read files.
C - Create	Allowed to create files. Unless they also have write access, they are not able to edit files that have been created.

W - Write	Allowed to make changes to files. Unless they also have create access, they may not be able to edit files since the write operation can only be used to extend files (not truncate them, which some file editors need to do).
E - Erase	Allowed to erase files and remove directories.
M - Modify	Allowed to modify file attributes.
F - File scan.	Allowed to see file and directory information. If a user does not have the file scan access right, they will not see any evidence of the file existing.
A - Access control	Allowed to change trustee rights. They are able to add other users as trustees, remove trustees, and grant/revoke specific rights from users. The only issue to be aware with for this right is that it is possible for users to remove themselves (as trustees) from directories, thus losing all access control!

In addition to trustees, groups and access rights, there is a concept of inherited rights. This means that users inherit rights from parent directories. For example, if user GEORGE has rights [CWEM] in a directory, and he has [RF] rights in the parent directory, then he will have [RCWEMF] rights as a result of inherited rights. This only works if one of the rights that GEORGE has in the two directories is granted to a group; if both are granted to him, he loses the rights of the parent directory.

Auditing and Security Tools

SECURITY is the standard security auditing utility, supplied with NetWare. It is found in the sys:system directory and needs to be run by a supervisor-equivalent user. Table 12 shows some of the most commonly used NetWare security tools and their uses.

Table 12. Common NetWare Security Tools

Application	Security-Related Uses
Security.exe	Creates a security report of trustee rights and account privileges.
Filer.exe	Most useful security feature is the “Who Has Rights Here” function. This let’s you see who has access to a particular directory or file.
Tlist.exe	This command displays trustees and their effective rights in the directory where the assignment was made. The syntax to use is: TLIST `directory`.
Syscon.exe	Intruder lock-out time restrictions and station restrictions (command user`list /`a to find out station addresses).
Login.exe	Make sure server/sys:login.exe is authentic. There are hacked versions of login.exe that can capture passwords. Check trustee rights in the login directory.

8.3.3.2 Novell NetWare Version 4.x

NetWare 4.x is not managed on a bindery-based security basis; it introduces a new layer called Novell Directory Services (NDS) security. With NDS, we move away from local user definition at each server and from user’s having to know where to find resources on the network. It could be regarded as the next step from the Windows NT Domain concept and is similar to the DCE and Warp Server Directory and Security Services (DSS).

NDS is the technology that provides a single, global view of all network services and resources. This allows users to access network services and resources with a single login, regardless of the user's location or the location of the resources. NDS offers a single point of control for an administrator through an easy-to-use graphical utility.

Key features of NDS:

- Global Access to Network Resources

  NDS is a global, distributed information database included with NetWare 4.x that provides access to all network resources, regardless of where they are physically located. It maintains information about every resource on the network, such as users, groups, printers, volumes, and other devices in a single, logical database.

- Hierarchical Directory Tree

  NDS works like a telephone book or an address book; it helps network users locate information by cataloging resources on the network and by organizing resources as objects in a hierarchical tree structure, known as the NDS directory tree.

  NDS establishes a rule-based administration, which allows administrators to grant access to an entire branch of the directory tree at once. This makes updating security access for an entire company fast and easy, minimizing the need to administer multiple groups as in previous versions.

- One Login, One Password

  NDS provides a central point of access to the entire network. Instead of logging in to many individual file servers, users and administrators log in to the network once, using one password. Access to resources after initial login is handled automatically through background authentication, this means that users are never asked for another password and are unaware that authentication has taken place.

- Distributed and Replicated

  NDS is a distributed database that is fully replicated to provide fault-tolerant login and administration from anywhere in the network. By breaking the NDS database into manageable pieces (partitioning) and distributing it across the network, fault tolerance is provided.

  NDS partitions are copied or replicated across the entire network as many times as necessary. If the primary partition is lost, the network instantly reconfigures itself to use a backup copy.

- Extensible

  The structure of an NDS directory tree is regulated by the directory schema. The schema is a set of rules that define how the NDS directory tree is structured. The schema determines which objects are defined, what attributes can be associated with objects, what properties objects inherit, and what positions objects occupy in the directory tree.

NDS Authentication Services

Because Novell Directory Services (NDS) replicates and distributes information across the network, this information must be kept secure from eavesdropping or tampering. Authentication services verify the validity of a user for each login and for each attempted access to other network resources.

NDS authentication services use Rivest-Shamir-Adleman (RSA) public/private key encryption technology.

RSA is a cipher based on the concept of a trapdoor function. This is a function which is easily calculated, but whose inverse is extremely difficult to calculate. In the RSA case, this function is factoring.

NDS authentication encryption allows authentication information to be transmitted in unreadable forms. Only during your initial login (user ID and password exchange) are you aware of authentication. Ongoing (background) authentication is transparent to you and takes place when you access other resources.

Note

Only information related to authentication is encrypted by NDS.

Public and Private Key Encryption

Public and private key, or asymmetrical, encryption uses two different keys that have a mathematical relationship to each other:

- Public key: This key is assigned to an object and can be published openly to anyone wanting to send a message to that object.
- Private key: This key is assigned to the same object, which keeps it secret. In public and private key encryption, either key can be used for encryption, and the other is then used for decryption. Because the mathematical relationship between the keys is difficult to compute, the public key can be made available freely without the risk of exposing the private key.

If you have the public key, you can encrypt messages to a client that owns the matching private key. You cannot use the public key to decrypt messages that have been encrypted with the public key. Only the private key can decrypt the messages encrypted with the public key. Because only one object has the private key, only that object can decrypt your messages.

Conversely, a sender can encrypt data using the private key. The recipients of this message use the public key to decrypt the message. If the decryption is successful, the recipient can be sure that the message was encrypted with the corresponding private key. In this case, many entries can decrypt the message, but only the holder of the private key could have generated the message.

NDS uses this private and public key technology for authentication, specifically to protect the authentication messages and to validate your identity. Correctly sending an encrypted authentication message demonstrates that the sender knows the corresponding encryption key. Along with using public and private keys, NDS uses two other components important in encryption:

- Nonce value. To prevent intruders from capturing and replaying authentication messages, these exchanges use nonce values, which are random values coined and sent only once.
- Message digest (one-way hash). This is an irreversible operation that allows the receiving server to verify that the sender has certain information without that information being sent.

NDS Authentication Process

Authenticating in an NDS environment consists of two processes:

- User login. When you make a client login, an NDS server is used to establish your identity. No matter where you are logging in, NDS walks the tree until it finds a writeable copy of your user object. NDS then retrieves your client's private key.
- Background authentication. After logging in, you still need to authenticate, or establish your identity, to a server holding other network services. This is done through background authentication.

User Login

Before you can log in to the Directory, the following information must be stored in the system. Your client must store:

- Your password.
- Your Distinguished Name (DN).

NDS stores:

- Your Distinguished Name (DN).
- A 32-bit pseudo ID value for your user object. This is an object-specific random number used to blunt a dictionary attack during login.
- A hash based on your password and pseudo ID value.
- Your public RSA key.
- An encrypted value of your private RSA key.

Background Authentication

Essentially, your login and authentication are the same process, but they are handled differently. From your perspective, login asks you to enter a password and then contacts the server. The server returns your context, and you are ready to work on the network. From NDS's perspective, login requires you to enter the password, which NDS uses to encrypt the private key. After the login process has completed, your client has its private key and is ready to authenticate to other network services. NDS handles this process in the background. Authentication data is valid only during the current login session. The critical data used to create authenticated messages for you are never transmitted across the network in plain text. To authenticate to a server, the client needs the following information:

- Credential. The credential is a data structure made up of a validation period and other user identification information. The validation period determines how long your credential is valid.
- Signature. The signature is the result of encrypting the credential data with your private key. Your client calculates the signature by:
  1. Applying the MD2 hash function to the credential, obtaining the hash value.
  2. Encrypting the hash value with your private key using RSA encryption.

Constructing Proofs

Although your client authentication can use the credential and signature repeatedly for multiple connections, it needs a new proof for each connection. A proof ties your client's signature to both the current request and the current session, thus making each proof unique to the request it accompanies. The proof also makes it unnecessary to transmit the signature. Your client and server need the following values to construct and verify a proof:

- Your user object's public key

- The number of bytes
- A proof order parameter
- The signature - a byte string treated as a large number with the most significant byte first. The signature is the central value in the computation. Your client needs your private key to calculate the signature. The server cannot calculate the signature, but can use the public key to determine whether your client holds the signature.

Packet Signature

Since authentication in NDS is session oriented, your client's signature (the basis of authentication) is valid only for the duration of the current login session. The signature itself is never transmitted across the network. Thus authentication services between servers and your client can be mutual and ongoing. Both sides of a session can be required to authenticate themselves, and the authentication process can proceed throughout the session. The NetWare Core Protocol (NCP) packet signature is a security feature that protects servers and clients that communicate through the NCP. NCP packet signature prevents packet forgery by requiring the server and your client to “sign” each NCP packet. The packet signature changes with every packet. When NCP packet signature is implemented both on the server and on your client, it is extremely difficult to forge a valid NCP packet.

General Concepts Introduced in NetWare 4.x Security

- Object Rights: These are used to manage and view objects within the NDS. There are five types of object rights:

Browse	The right to see an object in the NDS tree
Create	Enables the creation of a new object below in the NDS tree
Delete	This right is required to remove an object from the NDS tree
Rename	Enable the name of an object to be changed
Supervisor	Enables full rights to an object and its properties

  By default, when a user object is created in the NDS tree, a browse right is given to the user object itself. This is so that the users can see themselves in the NDS tree.

- Property Rights: Each object in NDS tree has a set of properties. Each property has rights associated with it. These rights control what users or other objects can do to the property. There are five property rights:

Compare	The right to compare any value to a value of the property.
Read	Enables the user to read the value of the property; compare is implied.
Add or Delete Self	The right to add or remove itself as a value of property. With this right, the user cannot affect any other values of the property.
Write	The right to add, remove, or change the values of the property, add or delete self is implied.
Supervisor	Grants all rights to the property.

- Access Control List (ACL): It lists and controls who has rights to the object and what those rights are. The ACL property contains the trustee assignment and the Inherited Rights Filter. To change the trustee’s access to the User object, you would change the trustee assignment in the User object’s ACL.

- Inheriting Rights: In NDS, object and property rights flow down the tree, just like file systems do. When a right flows down the tree, this is known as an *inherited right*.
- Inherited Rights Filter (IRF): The IRF is similar to the file-system rights filtering concept. With it, rights can only be taken away and not gained. Using the filter, you specify which rights can be inherited; if a given right appears on the list, it can be inherited.
- Effective Rights: An object's effective rights are what controls its access to other objects and those object's properties. Basically, effective rights are those rights a user can exercise on a given object. The NetWare 4.x operating system calculates effective rights to an object whenever a user performs an action. A user's effective rights are composed of:
  - Rights explicitly granted to the user requesting access. This implies that the user is listed in the ACL.
  - Rights explicitly granted to an object to which the user requesting access is security equivalent.
  - Rights flowing down through inheritance from container objects that have any of the explicit assignments in the bullets listed above.

  For example, in Figure 72, if user Y requests access to modify a Profile object, one of the following conditions must be true:

  - User Y has a trustee assignment to the container above the Profile object.
  - User Y is security equivalent to an object that has a trustee assignment to the container above the Profile object.

  The rights from that trustee assignment then flow down the Profile object through inheritance. These rights are included in User Y's effective rights.

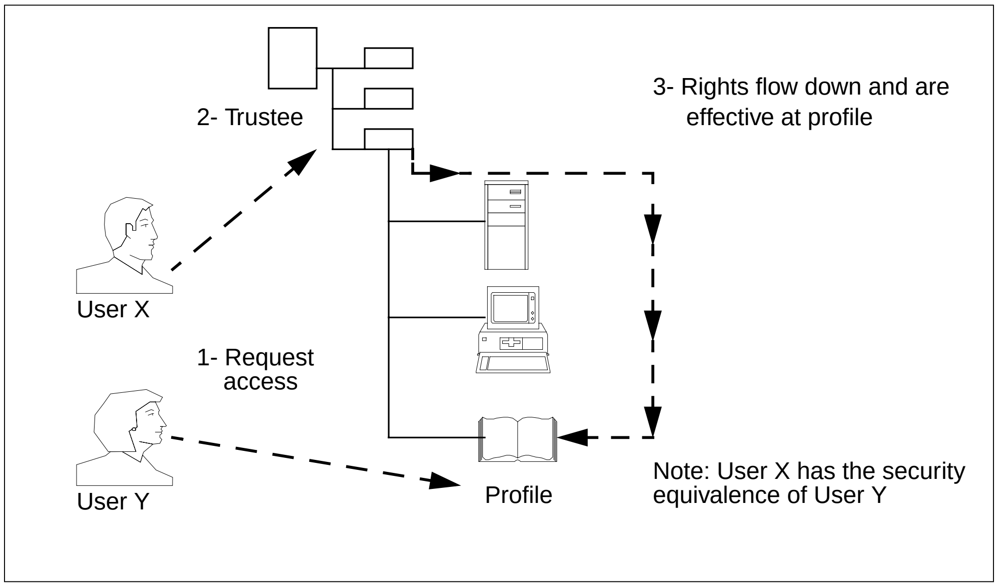

2- Trustee
3- Rights flow down and are effective at profile
User X
1- Request access
User Y
Profile
Note: User X has the security equivalence of User Y

Figure 72. Netware 4.x Effective Rights

- Administrative Access: When you create a new directory tree, a user named ADMIN is away created in the Organization (O) container and granted all rights to the tree and server file systems.

Note

User ADMIN does not have any special significance as the user SUPERVISOR did in previous versions of NetWare. It is only the first user object created. It is made a trustee of the [Root] object and given the Supervisor object right. This right effectively flows down to every object in the tree. Therefore, ADMIN has rights to create and manage all objects. The Supervisor object right can be filtered with an IRF.

8.3.4 Critical Novell NetWare Resources

The resources we must protect on NetWare are:

- SYS:SYSTEM
- SYS:ETC
- SYS:LOGIN
- AUTOEXEC.NCF
- Mail directories
- System Utilities
- Root-level directories of volumes

On NetWare 3.x, we also suggest to protect NET$OBJ.SYS, NET$PROP.SYS, and NET$BVAL.SYS.

Note

The most critical resource to protect on a NetWare environment is the physical location of the file server(s), and this should be a top security concern.

8.3.4.1 File Server Issues

Following is a list of issues to consider on the file servers:

1. Secure the File Server
   - Lock the console.
   - Lock the keyboard (Some file servers have key locks).
   - File server should be located in low traffic area.
2. RCONSOLE
   - Use a password different from the SUPERVISOR (ADMIN in NetWare 4.x) password.
   - Do not leave the password in the autoexec.ncf file.
   - Do not print out the password or system files that contain it.
   - Change the password periodically.
3. At the Server Console
   - Consider using SECURE CONSOLE.
   - Lock the console with MONITOR.
   - Stay alert to rconsole connections and the station addresses they come from.
4. Supervisor (ADMIN in NetWare 4.x) Account

   The Supervisor account is the most guarded and trusted account on the file server. It should be protected and monitored.

- Try not to work while logged in as Supervisor.
- Limit Supervisor (ADMIN in NetWare 4.x) equivalences.
- Limit concurrent Supervisor (ADMIN in NetWare 4.x) logins.
- Always log out when finished.

5. Watch for break-ins

    - Use conlog.nlm and examine console.log.
    - Look for intruder lockout notifications on the console.
    - Keep the system files invisible from users to prevent accidents.

6. File Protection

    - Keep data and applications separate.
    - Guard home directories with right restrictions.
    - Restrict access to sensitive areas on the server.

8.4 Functional Enhancements

TME 10 Security Management is likely to receive many more minor enhancements in future releases. Possible candidates are covered in this section.

8.4.1 Record-Level Subscription

Security resource records will support record-level subscription. By default the record-level subscribers are the same as the profile manager subscribers. For example, your profile manager's subscribers might be all the UNIX machines, but you might have one particular group record that you do not want to be applied on every one of those machines. You can edit the record-level subscriber list to remove unwanted machines. The other records in the resource will not be affected.

You may want to define different policies to apply to machines in perhaps a test environment, where password policies need not be as strict as in a production environment. You can do this by creating several records with different policies, either within the same or a different security profile. If the records are in the same system policy, you must tailor the subscriber list for each record to ensure that they are applied on the correct target systems. If you leave the subscriber list with the default values, which will be all the managed nodes and profile managers that subscribe to the profile manager in which the security profile resides, the last record that is received (which is not necessarily the last record in the list of records that you see on the screen) is applied to all the systems. If the records are in separate security profiles with different subscribers, you do not need to tailor the subscriber list at the record level.

8.4.2 Generic Access Permissions - RACF Update

Access permission names and values vary between operating systems. For example, the read access permission for a file is *r* on UNIX and *READ* on RACF. TME 10 Security Management provides a set of generic access permissions that are mapped onto the system-specific names when the records are distributed (see Section 5.10.1, "Generic Access Permissions" on page 86).

Once RACF is a supported endpoint, Table 6 on page 88 would look something like Table 13 below:

Table 13. Mapping Generic Access to Endpoint-Specific Values (Including RACF)

	Read	Write	Execute	Delete	Update	Control	Access	No Access
TACF								
FILE	Read	Write	Execute	Delete	Read Write Execute	chown chmod utimes sec Update		None
PROCESS							Read	None
CONNECT							Read	None
TCP							Read	None
TERMINAL							Read	None
SURROGATE							Read	None
NT								
FILE	R	W	X	D	Change RWXD	Full control		No access
DIRECTORY	R	Add			Change RWD	Full control		No access
SHARE	R				Change RWXD	Full control		No access
PRINTER						Full control		No access
RACF								
DATASET	READ	UPDATE	EXECUTE		UPDATE CONTROL	ALTER		NONE
GENERAL					UPDATE CONTROL	ALTER	READ	NONE

8.4.3 Security Record Population for RACF Endpoints

Refer to Section 5.12, “Populating the Security Profiles” on page 92, regarding TME 10 Security Management population of the profiles. The following is planned to occur for RACF when supported:

8.4.3.1 Security Group Records

A Security group for each RACF group is created during population. The RACF user membership is added to the Security group member list.

8.4.3.2 Security Resource Records

Population creates a Security resource for each RACF resource with the resource type of the Security resource based on the RACF resource class and the Security resource attributes based on the RACF resource’s security attributes.

8.4.3.3 Security Role Records

The populate option for security roles in RACF displays a dialog where you can specify user and group names from which to build the roles. It will:

- Search the RACF database for the occurrence of users or groups looking for resources that have those users or groups defined in the resources’ ACLs.
- Add the resource, resource type and access permissions to the new Security role’s endpoint-specific capabilities.

8.5 Integration with TME 10 User Administration

TME 10 Security Management and TME 10 User Administration are products that work with closely-related objects. The two development teams in Tivoli are working closely together to make links between the products where it makes sense. For example, the TME 10 User Administration password utility, wpasswd, does not link into the SeOS password checking (sepass). This means that someone using this utility would be able to set a password that did not conform to password restrictions defined in TME 10 Security Management. The current solution options include writing a script to call sepass and wpasswd if that is what is desired. Future versions of these products will integrate password checking.

A planned enhancement that could be available soon is the ability to automatically add a UNIX or NT user ID into a TME user profile from the TME 10 Security Management member list section of the security group record properties dialog panel. The name is specified in the same format as the user name field in a user profile, for example *George Bush*. You also specify the name of an existing user profile where the user will be created. The default policies of that user profile are used when creating the user. Any changes that you want to make to the user should then be performed through TME 10 User Administration.

Automatically creating a user in a TME user profile and making it a member of the Security group creates a dependency on the order of distribution of the profiles. If the user profiles and security profiles are in the same profile manager and distribution is performed at the profile manager level, the user profile is added before the security profile. In any other cases, the administrator should ensure the correct order of distribution.

Appendix A. General Security Concepts and Terminology

This Appendix explains some terms, concepts and techniques used to create secure environments for computer systems to cooperate over what are generally insecure communication links.

A.1 Terminology Used

To obtain a better understanding of data integrity in a secure environment, we must first look into some common terms used.

- Hash function: a process in which a secret key is used to generate a fixed-length output (called the message digest) from a variable-length input (the message to be secured). The message digest is appended to the message prior to transmission.
- MD5 (message digest 5): a secure one-way hashing function that converts an arbitrarily long data stream into a digest of fixed size.
- One-way hash function: a one-way transformation that converts an arbitrary amount of data into a fixed-length hash. It is computationally hard to reverse the transformation. MD5 is a clear example of a one-way hash function.
- Message authentication code (MAC): a cryptographic checksum value calculated by passing an entire message or the authentication elements of a message through an encryption mechanism. This MAC is based on a so-called symmetrical cryptographic algorithm (DES algorithm).

  For example, in an electronic data interchange (EDI) transaction, the sender calculates the MAC and appends it to the message prior to transmission. The receiver's software recalculates the MAC upon receipt of the EDI message and compares it to the original MAC. If the MACs agree, the EDI message is processed. If they do not agree, something in the message has been changed.

  This technique is used to identify whether messages have been modified, deleted or added.
- Symmetric cryptography: an encryption mechanism that uses the same key for encryption and decryption, sometimes referred to as secret key cryptography.
- Cryptography: the use of a suite of technologies designed to disguise the contents of a message. Cryptography typically involves an encryption operation to disguise the data at the transmitter and a decryption operation to remove the disguise at the receiver.
- Checksum: a computed value that depends on the contents of a block of data and which is transmitted or stored along with the data in order to detect corruption of the data. The receiving system recomputes the checksum based upon the received data and compares this value with the one sent with the data. If the two values are the same, the receiver has some confidence that the data was received correctly. The checksum may be 8 bits (modulo 256 sum), 16, 32, or some other size. It is computed by summing the bytes or words of the data block and ignoring any overflow. The checksum may be negated so that the total of the data words plus the checksum is zero.
- Cyclic Redundancy Check: a number derived from, and stored or transmitted with, a block of data in order to detect corruption. By recalculating the CRC

and comparing it to the value originally transmitted, the receiver can detect some types of transmission errors.

- Digital signature: utilizes public key cryptography and one-way hash functions to produce a signature of the data that can be authenticated and is difficult to forge or repudiate.

A.2 Authentication

Authentication is the process of identifying an individual, usually based on a username and password. In security systems, authentication is distinct from authorization, which is the process of giving individuals access to system objects based on their identity. Authentication ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual. You could say that the major function of TME 10 User Administration is to manage authentication systems, whereas TME 10 Security Management's major function is to manage authorization.

The main purpose of authentication is to identify, with near 100 percent certainty, the user of a protected resource. Most of the time, it relies on cryptography-based procedures, such as digital signatures. Authentication in a digital setting is a process whereby the receiver of a digital message can be confident of the identity of the sender and/or the integrity of the message. Once authentication is successful, the recipient of the request checks whether the sender has sufficient privileges to access the resource or perform the requested method. This is called access control.

In an object oriented framework, before an object can determine that a request is authorized, the object must know with certainty who is making the request. Identifying this request is called authentication, and a number of secure authentication protocols exist for this purpose. One of them is Kerberos (see "Kerberos" on page 191).

The need for authentication is not unique to object-oriented systems because it is also required for secure operation in client/server applications. What is unique, however, about distributed object environments is the large number of active entities making requests. In a client/server application, each client program can be viewed as a proxy (a mechanism whereby one system *fronts for* another system in responding to protocol requests) for some principal or user of the system. Authentication protocols assume that each of these principals is registered with the authentication system and has his or her own private keys and passwords. In an environment with thousands of objects scattered across hundreds of machines, it becomes difficult to manage objects as individual principals (a principal is an entity whose identity can be authenticated).

We must understand that communication between programs and users on a network opens a window of opportunity for persons who break into networks with the clear intention of doing damage. Intruders and vandals most of the time try to circumvent controls for authentication and authorization.

A.3 Authorization

Authorization or access control refers to the process by which one verifies or determines whether a principal is allowed (has the rights) to perform a specific operation. It takes place after the principal has been previously authenticated by the verifier.

The main purpose of authorization is to ensure that only those entities (known as initiators) with the necessary authorizations are allowed to access sensitive information and system resources (known as targets).

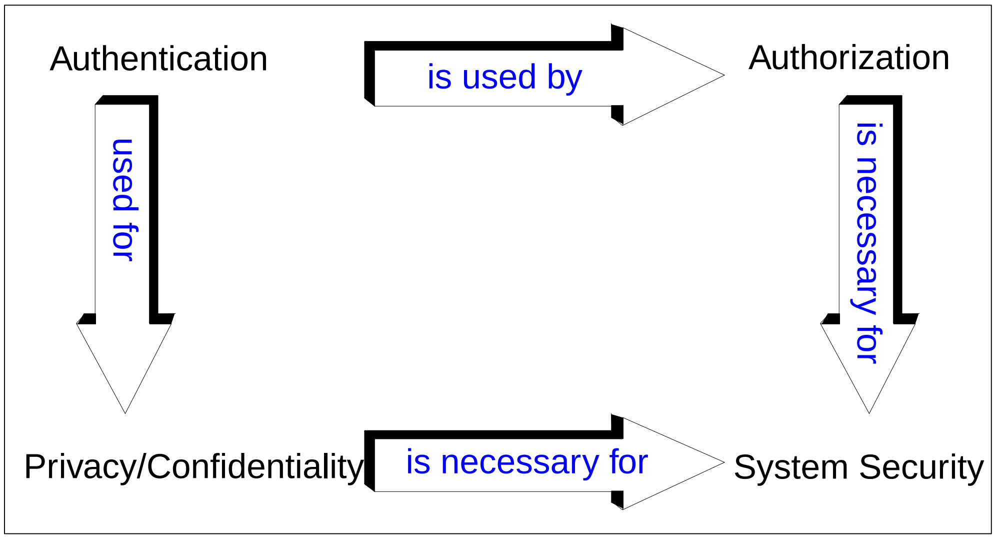


Figure 73. Relationship between Authentication and Authorization

A.4 Kerberos

Kerberos is an authentication service developed at Massachusetts Institute of Technology (MIT). Its primary purpose is to allow users and services to authenticate themselves to each other. Basically, it allows them to demonstrate their identity to each other.

Kerberos adds an extra layer of security to networks by registering users and services as Kerberos principals. When Kerberos users log in as Kerberos principals, they are asked to enter their Kerberos password, which, if correct, authenticates them to the Kerberos server. Kerberos then issues the user a ticket, which has a defined lifetime for authenticating the user to remote network services.

Kerberos works in a similar way to a personal identity card that has a photograph, name, address, and some other personal information such as a birth date or medical restrictions. The card has a limited lifetime, as represented by the expiration date. Kerberos works basically the same way, except that it's typically used when a user on a network is attempting to make use of a network service, and the service wants assurance that the user is who he says he is. After a user is authenticated to the Kerberos server, the user's application can request a session ticket to use a service. Kerberos creates a ticket that includes a session key and encrypts it with the service’s private key. The user’s application passes this ticket to the desired server. The server can decrypt it, and if its content is valid, access is granted to the client. The client/server session now uses the session key assigned by Kerberos. To that end, the user presents a ticket that is issued by the Kerberos authentication server (AS), much as a personal identity card is issued by an organization. The service then examines the ticket to verify the identity of the user. If all checks out, the user is accepted.

Therefore, this ticket must contain information linking it unequivocally to the user. Since the user and the service don’t meet face to face and the photograph on the card is of no use in this case, the ticket must demonstrate that the bearer knows something only its intended user would know, such as a password.

Note

Kerberos authentication fails for a user whose ticket has expired. The principal then is no longer trusted and is unable to perform additional work until a new ticket is acquired.

For additional information regarding Kerberos, please refer to Chapter 9 of the *TME 10 Framework Planning and Installation Guide Version 3.1* or the Request For Comments (RFC) 1510 on URL `http://www.cert.dfn.de/eng/resource/rfc/`.

A.5 Data Integrity

Let’s consider data integrity as the reasonable assurance that data has not been altered while en route from a sender to its intended recipient, and vice versa.

Integrity is the way of knowing how to preserve objects and how to make them trustworthy (for example, how to avoid the unauthorized modification of objects). Unauthorized users might not be able to read the contents of an object, but the protection system must be able to prevent an authorized entity from adding or modifying any parts of the object. Integrity of a resource is the issue of how to preserve the resource and how to protect it from unauthorized modification.

With this in mind, we must look into data integrity services as a way to stop unauthorized tampering with, or accidental corruption of, sensitive data.

To ensure data integrity, the sender of a message uses a secret key hash function against the message to generate a fixed-length appendix called a message digest. The message digest is sent to the recipient along with the message. Using the same key as the sender, the recipient applies the hash function to the message in an attempt to generate the same message digest. If the message digests match, an assumption is made that the data was not altered in transit.

To obtain a better understanding of data integrity issues, we must first define more common terminology used.

The use of a secret-key-based hash function to ensure data integrity is a very common practice. We now discuss some of these hash functions:

MD2, MD4 and MD5
These are cryptographic checksum algorithms that take as input a message of arbitrary length and produce as output a 128-bit “fingerprint” or “message digest” of the input. It is computationally impossible to produce two messages having the same message digest or to produce any message having a given prespecified target message digest.

MD2 is the slowest of the three. MD4 is the fastest and is part of the Simple Network Management Protocol (SNMP) Version 2 internet standard. MD5 is very similar to MD4, but has a more conservative design than MD4. It gives up a little bit of the speed of MD4 and replaces it with tighter security.

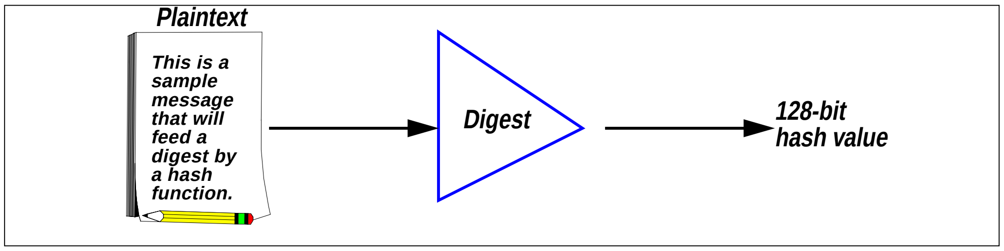

Plaintext
This is a sample message that will feed a digest by a hash function.
Digest
128-bit hash value

Figure 74. One-Way Message Digest Hash Function

Message digest functions are called *one-way* because knowing the message digest, one cannot reproduce the original message. Encrypted message digests give rise to integrity-protected messages.

Secure Hash Standard (SHS)
This standard specifies a Secure Hash Algorithm, SHA-1, for computing a condensed representation of a message or a data file. When a message of any length is input, the SHA-1 produces a 160-bit output called a message digest.

This message digest can then be input to the Digital Signature Algorithm (DSA) which generates or verifies the signature for the message. Signing the message digest rather than the message often improves the efficiency of the process because the message digest is usually much smaller in size than the message. The same hash algorithm must be used by the verifier of a digital signature as was used by the creator of the digital signature.

For additional information regarding these secret-key hash functions, please refer to Request For Comments (RFCs) 1115 (MD2), 1186 (MD4), and 1321 (MD5). They can be obtained from URL `http://www.cert.dfn.de/eng/resource/rfc/`. For SHS, please refer to URL `http://www.itl.nist.gov/div897/pubs/fip180-1.htm`.

A.6 Encryption

Encryption is the transformation of data into a form unreadable by anyone without a secret decryption key. Its main purpose is to ensure privacy by keeping the content of the information hidden from anyone for whom it is not intended.

Encryption works by transforming a message (called plaintext) into another message (called ciphertext) using a mathematical function and a special encryption password, called the key. Most encryption systems use the same key for both encryption and decryption (the process of converting the encrypted message back into plaintext).

Encryption keys are very similar to computer passwords; you need the key to gain access to the information stored inside an encrypted file. The encryption program uses the key to transform the ciphertext back into the plaintext. If you provide the correct key, you get back your original message; if you don’t, you get garbage.

Encryption, when used in communications, is the manipulation of a packet's data in order to prevent any but the intended recipient from reading that data. There are many types of data encryption, and they are the basis of network security.

There are two basic kinds of encryption systems:

- Private key encryption: It uses the same key to encrypt and decrypt the message.
- Public key encryption: It uses one key to encrypt the message and another key to decrypt the message. Public key means that you make one of the keys public without compromising the secrecy of the message or the other key.

DES

DES is the Data Encryption Standard, an encryption block cipher defined by and endorsed by the U.S. government in 1977 for use within the United States. It was originally developed at IBM.

The DES is basically a bit permutation, substitution, and a recombination function performed on blocks of 64 bits of data and 56 bits of key (eight 7-bit characters). The 64 bits of input are permuted initially and are then input to a function using static tables of permutations and substitutions. The bits are permuted in combination with 48 bits of the key in each round. This process is iterated 16 times (rounds), each time with a different set of tables and different bits from the key. The algorithm then performs a final permutation, and 64 bits of output are provided.

For additional information regarding the Data Encryption Standard (DES), refer to URL `http://www.itl.nist.gov/div897/pubs/fip46-2.htm`.

Appendix B. Introduction to TME 10 Framework Version 3.2

This Appendix gives you a brief introduction to Version 3.2 of the TME 10 Framework, also known as LCF for its implementation of the Lightweight Client Framework.

As we know, in Version 3.1 of the TME 10 Framework, a TMR server can serve up to 200 managed nodes (clients), and in order to handle more than 200 clients, you must install multiple TMR servers and connect the resulting TME 10 Management Regions (TMRs).

An enhancement to the TME 10 Framework, due for imminent release at the time of publication, is called the *Lightweight Client Framework* (LCF). LCF increases the number of resources a TMR can handle and, at the same time, enables those resources to be used more efficiently. The LCF is installed on systems (UNIX or PC) that are not used in daily operations of managing the network but are, instead, involved with business-oriented desktop operations. The LCF enables the monitoring of local operations or the receiving of management distributions from TME profile managers.

Communications with an LCF client is handled by an endpoint gateway (software running on a managed node) instead of by the TMR server, thereby reducing computing demand on the TMR server.

In the LCF, each TMR server still supports up to 200 managed nodes, but each endpoint gateway can now support a far larger number of endpoints, which is in the order of 1000s.

8.6 LCF Architecture

The LCF is a highly scalable extension to the TME 10 Framework that provides core TME 10 Framework functionality for all PCs and desktops in the enterprise. The centerpiece of LCF is the dataless, small-footprint client.

The LCF architecture introduces the following new components to the Tivoli Management Environment:

- An endpoint (EP) is a machine running the LCF. An endpoint should be a machine that is not used in daily operations of managing a network, but is instead a machine used for desktop operations. An endpoint can be either a PC or a UNIX workstation. An endpoint does not maintain a Tivoli database; so it requires far fewer resources than a full managed node.
- An endpoint gateway runs in a full TME10 managed node and provides communications between a group of endpoints and the rest of the TME 10 environment. The gateway also has the Mdist repeater functionality built into it, enabling it to act as the fan-out point for distributions to a very large number of endpoints.
- The endpoint manager runs on the TMR server and maintains an endpoint list, which keeps track of every endpoint in the TMR. This list contains all information necessary to uniquely identify and manage the endpoints.

Using these three components, the hierarchical structure of the TME changes from the current two-tier (TMR server and managed node) to a three-tier structure

(Figure 75 on page 196). The PC managed node will eventually be replaced by an intelligent LCF client that is part of the CORBA framework, thus building the third tier in the TME 10 Framework.

The LCF supports the Dynamic Host Configuration Protocol (DHCP). The address of an endpoint is established when the endpoint connects to the endpoint gateway. If the address changes between connections, either because of DHCP or a change in network topology, a new address is established the next time the endpoint logs in to the gateway. This feature accommodates endpoints that change locations frequently, as it would with portable computers.

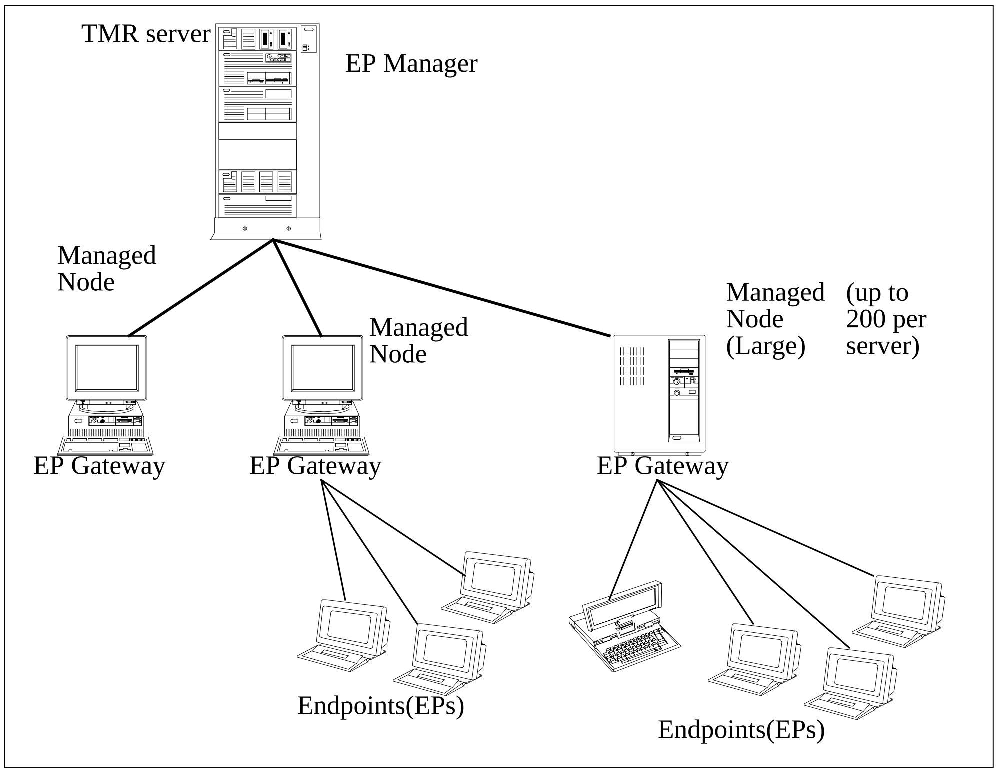


Figure 75. TME 10 LCF Enterprise Architecture

8.7 LCF Security

LCF will provide secure encrypted connections between the gateway and the endpoint. Endpoint-initiated actions will require principal login at the endpoint.

A TME 10 principal is another term used for the TME 10 administrator. To allow actions initiated from the endpoint, LCF will support principal login at the endpoint.

8.8 Endpoint Configuration and Initial Login

When the LCF daemon is first installed on an endpoint, the LCF daemon attempts to log in to its gateway. Because the gateway isn't known yet, the LCF daemon issues a broadcast requesting a gateway. For initial installation, this broadcast is ultimately forwarded to the TMR server.

When the TMR server receives an endpoint’s broadcast requesting a gateway, it consults the endpoint list to see which gateway the endpoint belongs to. The server uses a site-specific policy to select a gateway. For example, you may create a TME policy script that specifies that an endpoint in subnet 29 uses gateway A, while any endpoint in subnet 30 uses the gateway named B. The server then informs the gateway of its new endpoint; the gateway acknowledges the endpoint, and the login completes.

On subsequent LCF client reboots, when the endpoint knows its gateway, the LCF daemon attempts to log in to its gateway. When the gateway receives the login request, it services the request, and the login is complete. As a general rule, if a gateway hears a broadcast or a login request from an endpoint that it recognizes, the gateway services the request itself.

If the gateway does not respond to the endpoint’s login for any particular reason (for example, because it is down), the LCF daemon tries to contact other known gateways. If none respond, the LCF daemon issues a broadcast requesting a gateway, and this request is ultimately forward to the TMR server. The TMR server selects an endpoint gateway, and the login completes. If a gateway is down, the server selects a new gateway by using a policy that controls the endpoints.

After a successful login, both the endpoint and the gateway have a working channel by which to address one another.

Appendix C. Special Notices

This publication is intended to help anyone interested in managing security in the networked computing environment to understand and evaluate the contribution TME 10 Security Management has to make in this field. The information in this publication is not intended as the true specification of any product. See the PUBLICATIONS section of the IBM Programming Announcement for the TME 10 Framework, TME 10 Security Management or TME 10 User Administration, for more information about what publications are considered to be product documentation.

References in this publication to IBM (and/or Tivoli) products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (“vendor”) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

Appendix D. Related Publications

The publications listed in this Appendix are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

D.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see "How To Get ITSO Redbooks" on page 203.

- *Getting Started with TME 10 User Administration*, SG24-2015
- *TME 10 Framework V3.2, LCF and More*, SG24-2025 (in press)
- *An Introduction to Tivoli's TME 10*, SG24-4948-01 (in press)

D.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RISC System/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RISC System/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection (available soon)	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection (available soon)	SBOF-7250	SK2T-8042

D.3 TME 10 Information on the World Wide Web

A great deal of information, including access to IBM redbooks on Tivoli products, is available from Tivoli's home page on the web:

```
http://www.tivoli.com
```

How To Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at URL `http://www.redbooks.ibm.com`.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** – to order hardcopies in United States
- **GOPHER link to the Internet** – type `GOPHER WTSCPOK.ITSO.IBM.COM`
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get lists of redbooks:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Home Page on the World Wide Web**

  `http://w3.itso.ibm.com/redbooks`

- **IBM Direct Publications Catalog on the World Wide Web**

  `http://www.elink.ibmlink.ibm.com/pbl/pbl`

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** – send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to `announce@webster.ibmlink.ibm.com` with the keyword `subscribe` in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** (Do not send credit card information over the Internet) – send orders to:

	IBMMAIL	**Internet**
In United States	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** – send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** – send orders to:

United States (toll free)	1-800-445-9269
Canada	1-800-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1) 408 256 5422 (Outside USA)** – ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** – send note to `softwareshop@vnet.ibm.com`

- **On the World Wide Web**

Redbooks Home Page	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to `announce@webster.ibmlink.ibm.com` with the keyword `subscribe` in the body of the note (leave the subject line blank).

IBM Redbook Order Form

Please send me the following:

Title	**Order Number**	**Quantity**

First name	Last name	
Company		
Address		
City	Postal code	Country
Telephone number	Telefax number	VAT number

☐ Invoice to customer number ____________________

☐ Credit card number ____________________

Credit card expiration date	Card issued to	Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

DO NOT SEND CREDIT CARD INFORMATION OVER THE INTERNET.

List of Abbreviations

ACL	Access Control List
ADE	Application Developer's Environment
AMS	Application Management Specification
BDC	Backup Domain Controller
CORBA	Common Object Request Broker Architecture
DAC	Discretionary Access Control
DCE	Distributed Computing Environment
DES	Data Encryption Standard
DHCP	Dynamic Host Configuration Protocol
DN	Distinguished Name
DSA	Digital Signature Algorithm
EDI	Electronic Data Interchange
EP	Endpoint
ES1	Enhanced Scalability Release 1
GEM	TME 10 Global Enterprise Manager
GID	Group Identifier
GUI	Graphical User Interface
IBM	International Business Machines Corporation
IOM	Interobject Messages
IRF	Inherited Rights Filter
ITSO	International Technical Support Organization
JES	Job Entry Subsystem
LCF	Lightweight Client Framework
MAC	Message Authentication Code
MAC	Mandatory Access Control
MVS	Multiple Virtual Storage
NDS	Novell Directory Services
NFS	Network File System
NIS	Network Information System
NLM	Netware Loadable Module
NLS	National Language Support
OMG	Object Management Group
PDC	Primary Domain Controller
RACF	Resource Access Control Facility (OS/390 Security Server)
RPC	Remote Procedure Call
SAF	Security Authorization Facility
SAM	Security Account Manager
SHS	Secure Hash Standard
SID	Security Identifier
SNMP	Simple Network Management Protocol
SVC	System V Communication
TACF	Tivoli Access Control Facility
TCB	Trusted Computing Base
TEC	Tivoli Enterprise Console
TME	Tivoli Management Environment
TMR	Tivoli Management Region or TME 10 Management Region
UID	User Identifier

Index

ITSO Redbook Evaluation

TME 10 Security and Security Management
SG24-2021-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redeval@vnet.ibm.com

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction __________

Please answer the following questions:

Was this redbook published in time for your needs? Yes___ No___

If no, please explain:

____

____

____

____

What other redbooks would you like to see published?

____

____

____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

____

____

____

____

____