This is a draft version of a security manual, written by the Security RDS group in 1992 at Forge Drive. It took the group of some 20 attendees two weeks to prepare this information.
Caused by reorganizations within Tandem, we never finished the book.

But the good news is: It still is there – and here!

As of today in the year 2003, some chapters have to be deleted, re-written, or introduced.

This is an ongoing process and we plan to make the document as complete as possible within the next few months.

Updates will be available without notice from this place.

For the review team, which consists of some of the original writers,

Carl Weber (Carl.Weber@GreenHouse.de)

System Software Library

-DRAFT-
Revision 1.5
October 9, 1992

GUARDIAN 90™

Audit Guide

Release Version	GUARDIAN 90 C30
Product Version	SAFEGUARD C30
Date	October 1992

TABLE OF CONTENTS

Page

SECTION 1: INTRODUCTION

SECTION 2: GENERAL OVERVIEW OF THE C&L MODEL

Security Controls: How Much Is Enough?
Defining the Target Level of Security - Risk Assessment
Security Building Blocks
Control Categories
Least Necessary Privilege
Monitoring

SECTION 3: GUARDIAN OVERVIEW

Hardware
The NonStop Architecture and Fault Tolerance
The Operating System
Defining Users in the Tandem Environment
Processes
Tandem Advanced Command Language
The OLTP Environment
PATHWAY
ENSCRIBE
NonStop SQL
Transaction Monitoring Facility
Storage Media
INSPECT Debugging Tool
Security in the Tandem Environment
Guardian and Safeguard Security Features Comparison

TABLE OF CONTENTS

Page

SECTION 4: GUARDIAN SECURITY (WITHOUT SAFEGUARD)

System Security Protection

Basic Diskfile Security
Default Protection of Objects
Tape Security
Process Security
Pathway Controls
Clearing Deleted Files
Detection of Orphan Files
General Application Security Controls

Special Privileges

User Classification
User Authentication
Naming Conventions
The Super ID
Authorization to Define Security to Objects

Restricted Functions

Licensing
System Files

Extensions and Modifications

$CMON Generalized User Exit
PROGID Programs
INSPECT and DEBUG

System Configuration

Initializing the Security Environment
SYSGEN Parameters
Auditing Attempts to Access System Resources
Database Audit Trail
Auditing System Console Activity
Password Management Features
Encryption of Passwords

TABLE OF CONTENTS

Page

Network Security

Identifying Users of the EXPAND Network
Common EXPAND Network Capabilities
Dial-Up Access Concerns
Deleting Network IDs
Protection of Network Devices
Remote System Maintenance
Spooling Facilities
Other Tandem Network Security Considerations

SECTION 5: SAFEGUARD SECURITY

System Security Protection

Basic SAFEGUARD Resource Security
Additional Protection Attributes
Default Protection of Objects
Explicit Access Denials
Freezing System Resources
SAFEGUARD Protection Records
Clearing Deleted Files
Temporary Users

Special Privileges

The Super ID
Authorization to Define SAFEGUARD Security
Special SAFEGUARD Groups

Restricted Functions

System Files

TABLE OF CONTENTS

Page

System Configuration

Bringing Up the Security Environment
Controlling the Logon Process
Auditing Attempts to Access System Resources
Auditing Logon Attempts
Auditing Access to User Authentication Records
Auditing Attempts to Access Objects
Auditing Access to Object Authentication Records
Recommended Level of Audit
Audit Record Management
Password Management Features
Encryption of Passwords
SAFEGUARD Conversion Issues
Authorization Search Sequence

Network Security

Identifying Users of the EXPAND Network

SECTION 6: AUDIT PROGRAM/CHECKLISTS

GUARDIAN Audit Program
SAFEGUARD Audit Program

APPENDIX A: ACCESS NEEDED TO REVIEW THE TANDEM ENVIRONMENT
Access Needed to Review the Tandem Environment
Key Commands

P R E F A C E

PURPOSE

This manual is designed to help auditors and security administrators responsible for Tandem NonStop™ Guardian 90™ system installations to plan and implement a sound security environment suited to their business requirements. This manual emphasizes the security capabilities of the Tandem Guardian 90 environment, establishing an appropriate security standard based upon those capabilities, and methods and procedures for monitoring and auditing compliance to those standards.

TARGET AUDIENCE

This audit guide is designed primarily for use by auditors and security administrators responsible for the protection of business information and processing within Guardian 90 systems. A basic understanding of the Guardian 90 environment should be obtained before using this manual. This audit guide contains both overview Specific topics that may be useful to computer installation managers, computer operations staff members, and system programmers.

CONTENT AND STRUCTURE OF THIS AUDIT GUIDE

The Tandem Security and Audit Guide includes the following topics:

Section 1:	Introduction
Section 2:	General Overview of the Security Audit Model provides highlights of general security concepts
Section 3:	GUARDIAN Overview outlines recommended strategies for monitoring security within a Tandem Environment
Section 4:	GUARDIAN Security (Without Safeguard) provides a description of the security environment.
Section 5:	Safeguard Security discusses the additional elements in this environment.
Section 6:	Audit Program/Checklist contains a checklist based on the issues discussed in previous sections that can be used as a tool for auditing the Tandem environment.

USING THIS GUIDE AS AN AUDIT MANUAL

This guide describes the security features of Tandem Nonstop systems. For each feature, the recommended implementation is described, along with the implications of the features. Finally, the audit features are discussed, providing the auditor guidance in evaluating the effectiveness of controls surrounding the security feature.

Examples of commands are provided in the following chapters. Commands are designed to display the desired information on a Tandem terminal. The auditor may wish to route the command output to a printer in order to facilitate his or her analysis. APPENDIX A: ACCESS NEEDED TO REVIEW THE TANDEM ENVIRONMENT discusses the security authority needed to complete an audit, and summarizes the use of key online commands and utilities.

SUGGESTED READINGS

For additional information on Tandem Security or the Tandem NonStop Environment, the following manuals are helpful:

Introduction to NonStop System Operations
SAFEGUARD Reference Manual
SAFEGUARD Administrator's Manual
Security Management Guide
GUARDIAN Programming Manual

ACKNOWLEDGEMENTS

This document is the result of a joint effort between Tandem Computers and Coopers & Lybrand. The effort was intended to provide a methodology for auditing security in a Tandem environment. The participants in the effort were:

William Buer - Tandem Computers Incorporated

Matt Matthews - Tandem Computers Incorporated

James Stibbards - Tandem Computers Incorporated

Carl Weber - Tandem Computers Incorporated

Michael Goldfarb - Coopers & Lybrand

Lawrence Jewik - Coopers & Lybrand

Kurt Meiser - Coopers & Lybrand

Xenia Ley Parker - Coopers & Lybrand

Ted Schaefer - Coopers & Lybrand

The studies were completed on a model NonStop EXT25 with version C30 of GUARDIAN 90 and version C30 of SAFEGUARD.

SECTION 1
INTRODUCTION

The methodology upon which this manual is based analyzes a security environment from the perspective of the following key categories:

- system protection
- special privileges
- restricted functions
- extensions and modifications
- system customization
- network security.

The basic elements of security may be grouped into one of these categories for analysis purposes. For each element in the Tandem Guardian 90 operating system environment, this guide describes the security feature of the element, followed by the recommended implementation, and how to audit the security feature.

Why Security Is Important

Online transaction processing and computer information systems have become critical to organizational productivity. In many industries, information is the key asset of the organization. All critical information should be protected and monitored, whether it is locked in traditional filing cabinets or stored inside computers.

An increasing awareness of computer security issues is evident in the media coverage of hackers, disruptive attacks by logic bombs and viruses, and increasing legal and regulatory requirements. There is a growing realization that good security measures support the objective of continuous system availability and can reduce the cost of unintentional errors, two of today's important operations and manageability issues.

Since the introduction of Tandem systems, we have seen increasingly larger and more complex operating environments for both business and government applications. Previously independent computer systems have grown into heavily interconnected networks of computers including combinations of Tandem's and other vendors equipment. These heterogeneous enterprise networks have become the way business is done to remain competitive. Tandem is committed to addressing the exposures and risks created by this rapid expansion and diversification in the operating environment.

Security and Management

The basic management issues surrounding computer security are similar to those in other areas of an organization; except management policy for computer systems and networks must be automatically enforced by access control and cryptographic security products. Traditional manual procedures are still needed to accomplish management control objectives in the business organization external to the computer, for example, physical security, employee relations, and application input-output reconciliation.

Computer security is like many other areas of a business where there are many conflicting objectives, such as profit, risk, accuracy, timeliness, productivity, and quality, that must be managed on a daily basis. Banks loan money, but also establish loan loss reserves. Manufacturing companies balance production rates against quality assurance rejects. Computer security has personnel errors, theft, viruses, impersonators using someone else's computer identity, and hackers. The common thread is that management participation is needed to determine the appropriate balance between the cost of protective measures, including auditing, administration, and end-user productivity, versus the loss when an incident occurs.
Implementation of successful computer security requires a balanced approach with attention to technology, people, cultural factors, and management judgement.

Tandem's contributions cover the full spectrum of information security issues:

- Confidentiality
- Integrity
- Availability

Tandem is the recognized leader in continuously available, fault-tolerant systems. Guardian Transaction Monitoring Facility™ (TMF) was the first product to ensure database integrity and recoverability during simultaneous OLTP, batch, and query processing. The Guardian Remote Duplicate Database Facility™ (RDF) was the first product to automatically provide a remote copy of critical production data to support business resumption within minutes of a major disaster.

Fundamental Security Services

Tandem has defined five basic security services that protect computer information and ensure compliance with legal and auditing requirements. Subsequent portions of the statement of direction will refer to these services.

Authentication ensures accurate user identification. User authentication is required for any security program. The concept of user as an accountable subject is a basic security principle. Accurate user identification may use a simple password or special user authentication devices, such as the hand-held Atalla Challenge Response unit or biometric identification devices. Higher business risk justifies more accurate, and correspondingly more expensive, authentication procedures to prevent imposters or hackers from entering the system.

Authorization automatically controls the access to resources. The authorization process continuously monitors and enforces a defined security policy. The access rules enforce information confidentiality, authorized information changes, creation of new information, and access to specific system services, subsystems, programs, processes, and devices.

Administration defines access rights and privileges of users. Administration is the translation of external business policy decisions into internally enforceable access rules throughout the computing enterprise. Administration also includes adding, changing, and deleting the employees and others who can use the organization's computing facilities. Administration is an ongoing activity due to ever changing external factors like, employee turn-over, reorganizations, new business policies, and legal requirements. Administration is the bridge between the business world and the electronic world of the enterprise computing network.

Auditing monitors user activity and access within the system. Monitoring is frequently carried out by three separate groups - security administrators who are responsible for implementing the security policy, internal auditors who ensure proper administrative and monitoring procedures are being followed, and external auditing firms, such as Coopers & Lybrand, that may annually review all activities. Auditing ensures management policies are being followed and the access rights and privilege definitions are correct. These three levels are intended to detect inappropriate or fraudulent activity by any individual in this management control hierarchy. The auditing function requires the existence of a thorough record of security administrative activities and security events within the system. The security auditing function identifies the user, the name of the object, and the access decision.

Cryptography protects the integrity and confidentiality of data. It ensures data is protected when transmitted across communication networks, stored off-site, and used within computer systems. This is accomplished through mathematical techniques that scramble the original message or detect even single bit modifications to data or computer programs. Cryptography also provides positive authentication of users and data objects.

NonStop Systems Overview

Tandem NonStop systems are primarily used for on-line transaction processing, message switching, database, and batch processing applications that require continuous availability and linear expandability. All NonStop systems ensure the continuous availability and integrity of transactions. NonStop systems are based on a loosely coupled architecture that consists of multiple processors, dual interprocessor buses, dual-ported controllers, and multiple power supplies. Should there be a power outage, system memory is preserved via battery backup modules. To further support continuous availability, components can be easily replaced on-line, without shutting down mission critical applications. Data integrity is integral to the design of NonStop systems, including hardware onboard error-correcting memory, and parity checking on data and instruction paths.

A single NonStop system with Guardian 90 can be expanded from 2 to 16 processors with linearly proportional throughput increases per processor. Tandem NonStop systems with Guardian 90 can be interconnected into large high performance central site complexes with up to 224 processors using Tandem's proprietary Fiber Optic Extension™ (FOX) and EXPAND™ interprocessor protocol. Geographically distributed networks of up to 4,080 processors can be created using EXPAND in conjunction with popular communication protocols, such as CCITT X.25, IBM SNA, and IEEE 802 LANS. EXPAND allows any mix of local or remote processors to appear as a single application processing system.

The NonStop hardware architecture and the Guardian 90 operating system architecture prevent a single hardware or software malfunction from disrupting system operations. In this parallel processing architecture, the workload is divided among the processors, which perform multiple tasks simultaneously. Under normal operation, all processors share the workload; there are no idle backup units. In the event one processor fails, the workload of that processor is automatically taken over by the others.

Tandem's system architecture allows for centralized or distributed databases. Data can reside on a single NonStop system or it can be distributed on multiple NonStop systems connected via fast local fiber optic Extension (FOX) interconnections or conventional communications lines. Data can also be distributed across wide area networks of NonStop systems by using the EXPAND product. Although data is distributed to various systems, a distributed database appears to users and applications as a single system.

The Guardian 90 operating system is designed to provide software fault tolerance that complements the hardware fault-tolerance of Tandem systems. A software or hardware failure in one processor does not disrupt processing, since Tandem NonStop systems use loosely coupled processors each with independent copies of the Guardian 90 operating system to redistribute the workload the other processors. The Guardian 90 operating system provides data integrity within a single system or across a FOX single system image or across a distributed network. Tandem's Transaction Monitoring Facility (TMF), which is a part of Guardian 90 operating system software, protects data from the effects of incomplete transactions, system failures, FOX failures, or network failures.

Tandem's Distributed Systems Management (DSM) services and applications enable effective operational monitoring and control of computer resources. DSM provides a single management system for collecting, logging, and distributing computer operations management data within a single system or FOX and EXPAND interconnected single system images of NonStop systems.

Tandem NonStop systems with Guardian 90 provide in-the-field upgrade ability, which is of particular importance to customers that anticipate rapidly changing workload requirements. This flexibility allows installations to increase computer system performance with minimum cost and service disruption. Tandem NonStop hardware construction allows technicians to complete many maintenance functions and hardware upgrades without removing the system from service. Service technicians can even field upgrade systems to accommodate unusual or emergency situations by increasing the processing capacity of production systems and networks by moving processors from development and testing systems.

NonStop Systems Security Overview

Tandem NonStop systems are high-performance, loosely-coupled parallel processor computer systems designed to provide continuous availability for online transaction processing, query, and batch processing. The combination of NonStop hardware architecture and the Guardian 90 operating system software provide fault tolerance and data integrity through the following features: multiple independent processors interconnected by dual high-speed internal buses, dual-ported disk drives and device controllers connected to different processors, multiple power supplies, fault-tolerant software process pairs in different processors, the Transaction Monitoring Facility (TMF) software, failed and run-away hardware and software error detection. All systems provide internal battery-backed memory and automatic power-on resumption of processing. NonStop systems allow online maintenance and repairs of virtually all failed components, including processors, I/O controllers, and power supplies, while the remainder of the system continues application processing. The Tandem Maintenance and Diagnostic System (TMDS) automatically monitors the status of the processors, I/O controllers, power systems, and cabinet environments.

The NonStop hardware architecture defines two processing states which separate privileged Guardian 90 programs from non-privileged user programs. The hardware also enforces virtual-to-real memory mapping, separation of program instruction pages from data pages, and separation of individual user process address spaces.

The Guardian 90 operating system combined with the Safeguard access control product provide security protection mechanisms to mediate access to system resources and prevent object reuse. Discretionary Access Controls (DAC) can be applied to the following object types: disk volumes, sub volumes, files, devices, sub devices, and program inter-processes communication. The access control lists can explicitly deny or allow access by specific individuals, by groups of individuals, or both. The system can be configured to protect only selected objects or all objects can be protected by default. All individuals, system operators, and security administrators are subject to DAC mechanisms.

Individual user authentication records are maintained by the Safeguard product in a protected file. Designated security administrators create, maintain, and delete user authentication records. Each user has a unique logon ID and password. Each logon ID can have a predefined expiration date or may be temporarily suspended from system usage. Safeguard software provides optional password management features for enforcing minimum length, periodic expiration, minimum change interval, 60-entry history, one-way encryption, invalid password attempts threshold, password expiration warning, and expired password grace period. Upon successful authentication it provides prior logon date and time notification.

The Guardian 90 operating system with the Safeguard product provides an auditing facility that records security-relevant events such as logon and logoff; file open, close and purge; new process creation; the use of operator privileges; and changes to the security policy. An audit reduction tool is provided to display activity by specific subjects, objects, and time ranges.

The Safeguard product supports three administrative roles to control the use of restricted security commands and definition of controls. The first two roles, defined by the security-administrator and security-operator groups, designate which individuals can use audit service commands, terminal commands, and alter options, as well as stop Safeguard product enforcement. The third role, defined by ownership rules and object-type authorization records, controls the creation and maintenance of subjects, objects, and access privileges. The Safeguard product can be configured to provide a complete audit trail of all administrative changes.

Guardian and Safeguard Security Features Comparison

The Safeguard access control security product is optional on Guardian systems. This is similar to the optional nature of RACF, CA-ACF2, and CA-TOP SECRET products in the IBM MVS operating system environment. SAFEGUARD runs as an extension of GUARDIAN security to provide additional, enhanced levels of protection.

The following two part table summarizes some of the differences in GUARDIAN and SAFEGUARD features.

Security Feature	**GUARDIAN**	**SAFEGUARD**
USERS		
User Authentication	Yes	Yes
Remote Password Authentication	Yes	Yes
Password Expiration	No	Yes
User-ID Expiration	No	Yes
Logging of Sign-on Attempts	No#1	Yes
Logging of Attempts to Modify Security Record	No	Yes
Minimum Password Length	Yes	Yes
Password Encryption	Yes	Yes
Prompt for Old Password Before Allowing Password Change	Yes	Yes

OBJECTS	**GUARDIAN**	**SAFEGUARD**
VOLUME/SUBVOLUME/DISKFILE		
Available Authorities	RWEP#2	RWEPCO
File Attributes (LICENSE, CLEARONPURGE, PROGID)	Yes	Yes
Logging of File Access	No	Yes
Logging of Attempts to Modify Security Record	No	Yes
OTHER DEVICES AND SUBDEVICES		
Available Authorities	None	RWO
Logging of Device Access	No	Yes
Logging of Attempts to modify Security Record	No	Yes
PROCESSES AND SUBPROCESSES		
Available Authorities	None	RWPCO
Logging of Process Name Access	No	Yes
Logging of Attempts to Modify Security Record	No	Yes
Control of NAMED or UNNAMED as a group	No	Yes

#1 May be implemented through $CMON, but only for TACL events
#2 Disk file only

SECTION 2
GENERAL OVERVIEW OF THE SECURITY AUDIT MODEL

This section discusses basic security concepts associated with many technical platforms. Section 4 will focus upon implementation of these concepts on a Tandem computer platform.

SECURITY CONTROLS: HOW MUCH IS ENOUGH?

Effective security benefits all users of a system since it ensures the confidentiality, integrity, and availability of the data. As the costs of securing processing environments has risen, organizations have struggled with the amount of resources to devote to security.

Assume that the degree of security controls in place can be measured on a scale of 1 to 10 with 10 representing the implementation of an ideal level of security controls such that no security risks are present. In all likelihood, implementing security controls at this level "10" would adversely affect areas such as system performance and availability, employee productivity, and various organizational resources. Thus, representatives from throughout the organization must decide upon the **target level of security** that adequately protects their informational assets, while maintaining the installation's availability and performance goals.

```
 ___          10    Maximum Security Controls in Place
  |
  |           <--Target Security
  |
  |
 _|_           5   "Average" Security Controls in Place
  |
  |
  |
  |
 _|_           0    NoSecurity Controls in Place
```

Ideally, determination of the level of protection is driven largely by the business requirements for confidentiality, integrity, and availability of its information.

DEFINING THE TARGET LEVEL OF SECURITY: RISK ASSESSMENT

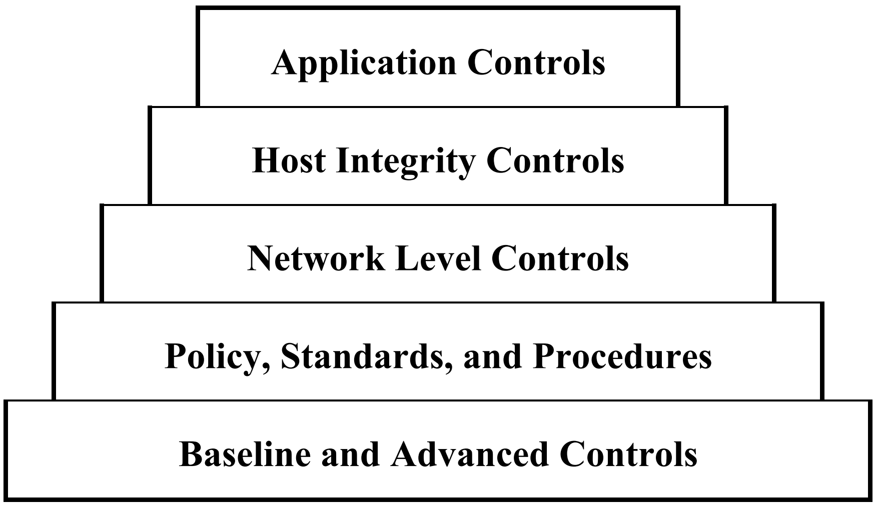
A tool which is often used to determine the level of protection required is the information security risk assessment. Security directors often discover that obtaining resources to secure informational assets can be difficult to cost justify based merely on additional protection. On the other hand, when the need for improved security can be directly related to **actual business risks,** the justification can be made easier.

During the risk assessment, the security organization works with the user groups which are vital to the organization being a going concern. Together, they determine the importance of data integrity, confidentiality, and availability to each user group's particular function in running the business. Once the user concerns, i.e. the business needs, are understood, the minimum level of controls can be identified.

In addition, the risk assessment exercise can be an outstanding means of promoting security awareness throughout the organization. Often, users do not consider the consequences of security breaches and how they can adversely impact business functions.

SECURITY BUILDING BLOCKS

A sound security environment is composed of a hierarchy of controls in the following areas:



Hierarchy Of Controls

Software security controls in general are dependent on a foundation of management controls comprised of a security policy, standards, and procedures. Within the hierarchy of software controls, the effectiveness of application controls depends on the existence of appropriate system integrity controls. In a networked systems environment, host integrity controls may be affected by existing network security controls.

As a result, security within any environment may be compromised if exposures exist on a lower level control, despite the effectiveness of the controls at the top of the structure. For example, the lack of host integrity controls places the entire security structure at risk, even when network controls are adequate. Thus, when planning the security environment, it is important not to neglect any of these building blocks of the security structure.

Application Level Controls

Application security controls in general can be described as a matrix of system users (subjects) and system resources (objects) containing access rules. However, even well designed application controls may not provide appropriate security if non-privileged users can bypass or freely alter access rules or if the number of privileged users is not reasonably limited.

Host Integrity Controls

Host integrity controls concern in general ensuring that a separation is maintained between general system users and the operating system. The effective implementation and maintenance of host integrity controls is required to ensure the effectiveness of access control software.

Host integrity controls concern the following primary areas:

- Ensuring that system privileges are assigned in a restrictive manner and that critical functions are restricted to trusted users,
- Maintaining system integrity through the appropriate selection of system customization parameters and the installation of system extensions and modifications which do not introduce integrity exposures,
- Protecting operating system resources such as files, tables, etc.

Network Level Controls

From a local security and integrity point of view, it is important that fundamental network controls be assessed and that the level of trust established with other nodes be verified. If trusted nodes do not have adequate system integrity, the local host's integrity may be exposed.

When systems are connected to a network, local integrity controls must be complemented by network controls for a number of reasons:

- A local host may be set up to rely on authentication performed at other nodes. Jobs or session requests may be accepted from "trusted nodes" without re-authentication such as without checking userid/password combinations,

- A local host may accept input data and update requests from "trusted nodes" without performing validity and authority checking that is usually done locally,

- A local host may transmit classified or highly critical information through the network, relying on the network to protect it during transmission and to route it to the proper receiving node. In addition, there is reliance on the receiving node to properly handle and protect the information.

Policy, Standards, Procedures

When systems are connected to a network, local integrity controls must be complemented by network controls for a number of reasons:

- A local host may be set up to rely on authentication performed at other nodes. Jobs or session requests may be accepted from "trusted nodes" without re-authentication such as without checking userid/password combinations,

- A local host may accept input data and update requests from "trusted nodes" without performing validity and authority checking that is usually done locally,

- A local host may transmit classified or highly critical information through the network, relying on the network to protect it during transmission and to route it to the proper receiving node. In addition, there is reliance on the receiving node to properly handle and protect the information.

Policy, Standards, Procedures

Security management is required to establish and maintain the required level of security. A security policy is needed to set and communicate the general rules and directions. Security standards translate the policy into technical standards, options, and settings for the operating system and the security software. Procedures and guidelines provide a detailed interpretation of the policy for the different roles related to security such as security administrators, resource owners, end users, and auditors.

**** NEED GRAPHIC FROM C&L ****

**** PREFER HARD COPY ****

(PREVIOUS PC GRAPHIC DID NOT CONVERT TO MS-WORD ON MAC)

Without adequate security management, controls are rarely built according to an organization's needs and are usually not kept at the required level. As depicted in the above figure security and integrity controls tend to deteriorate over time unless they are closely monitored.

Baseline And Advanced Controls

Security controls fall into two general categories -baseline controls and advanced controls.

Baseline controls are needed to establish a working set of controls; if one is defective or missing, the whole set is defective. Operating system integrity controls are a good example of baseline controls; if one control area is exposed, the whole system has an integrity exposure. An analogy to this in physical security is that a fence must be complete and all doors must have a reasonable lock to establish basic protection.

Advanced controls are controls that may be considered for additional security beyond baseline usually based on specific risks. In physical security, advanced controls would be increasing the height of a fence or installing extra strength locks on the doors.

Ultimately, the installation's security policy and standards must somehow address the implementation of the controls within each of the building block steps shown. Depending on the platforms in place at a given installation, the policy and standards may be structured in different ways. Defining security standards in terms of the categories defined previously - application security, access controls, operating system controls, and hardware controls, etc. may be appropriate for some platforms, but may not be applicable for others. On some platforms for example, access controls are actually a **part** of the operating system. As a result, defining the platform security standards in this format may be less than ideal.

CONTROL CATEGORIES

As a result, the framework on which this guide is based has been developed to be applicable to a wide range of processing platforms. This is particularly useful in today's multi-platform, multivendor environments.

In this framework, security controls can be grouped into the following categories:

System Protection
Special Privileges
Restricted Functions
Extensions and Modifications
System Customization
Network Security

These categories can be defined as follows:

System Protection - In this category of controls, the general protection of the system resources is emphasized.

Special Privileges - This refers to powerful, high-level system privileges granted to users based on the capabilities of the technical platform involved. Controls in this area tend to revolve around a single user or group of users.

Special privileges are the processing and access capabilities given to system users. Access may be provided to restricted system areas, high level privileges, and sensitive functions. Access in any processing environment should be based upon the concept of "Least Necessary Privilege". This concept provides access based strictly on need, as opposed to global access.

Restricted Functions - These include special system capabilities. Controls in this area focus on a system function as opposed to a user capability. These functions should generally be limited to a reasonably small set of users.

Extensions and Modifications - These refer to system code modifications that are made by the individual installation. These modifications are internally developed and installed, and are usually outside of the normal capabilities provided by the vendor. (Note that normally there are few of these in this environment.)

Security and integrity concerns may exist for two reasons:

1. The additional code, which may contain privileged instructions, may be poorly designed and contain unintentional exposures.

2. The code can be designed with good or malicious intent, containing exposures such as trap doors.

Therefore, it is essential that system extensions and modifications be tightly controlled, properly documented, and authorized by appropriate levels of management.

System Customization - Most systems have parameters or options which can be set to suit the individual needs of the installation using it. System customization controls focus on how these options are implemented.

Network Controls - Here, controls surrounding resources accessible via network access are examined.

By analyzing controls in these terms, a framework exists which is applicable across most platforms to control the security building blocks.

LEAST NECESSARY PRIVILEGE

The foundation of many of the principles discussed in this manual is based on the concept of "Least Necessary Privilege". This concept states that individuals should receive only the minimum number of system privileges that are required to complete their job responsibilities and provide an appropriate segregation of duties. Application of all the concepts and audit features described in this guide should be based on Least Necessary Privilege.

MONITORING

The methodology described in this manual stresses the importance of monitoring controls on an ongoing basis to ensure that the security standard achieved is maintained over time. Depending on the control areas addressed, some monitoring may take place on a daily basis. Other areas might require less frequent intervals, such as weekly or monthly monitoring.

If it is determined that monitoring of the environment has taken place at adequate intervals by qualified individuals with appropriate segregation of duties, monitoring results should be utilized as much as possible in the auditing process.

SECTION 3
GUARDIAN OVERVIEW

The auditor should obtain a basic understanding of the Tandem environment from sources other than this guide. However, this section provides a description of basic features of the Tandem NonStop environment.

HARDWARE

The computers in the Tandem family of NonStop systems include the Tandem Cyclone, VLX, TXP, TNS II, EXT, and CLX systems. This guide focuses on security features of those systems. Tandem local area networks and Tandem UNIX systems are not covered.

THE NONSTOP ARCHITECTURE AND FAULT TOLERANCE

The Tandem NonStop architecture provides capabilities for fault tolerant processing, or continuous availability of the computing system. This fault tolerant concept extends to hardware, software, and network facilities and is generally transparent to general system users.

Programs known as process pairs run in different processors, where a process pair consists of a primary process and a backup process. Each processor consists of its own Instruction Processing Unit (IPU), main memory, I/O Channel, interprocessor bus, and power supply, thus making each processor capable of operating independently of other processors in the system.

*** Figure 8-5: Components of One Processor Module ***
*** Introduction to Tandem Non-Stop Systems, S8031-049, p/n 82503 ***

The primary process executes in one processor, while the backup remains in another processor ready to resume activity should the primary process fail. The DYNABUS is the dual interprocessor bus that is used by the processors to communicate. It is through the high speed DYNABUS that the primary process sends periodic checkpoint messages to the backup process as to the status of processing. If the primary process fails for any reason, then the backup process can resume processing.

Fault tolerance for the database is achieved by establishing two different paths between a processor and storage device. In addition, the NonStop environment allows for mirrored disks, which are different disk volumes that contain identical copies of the database. The mirroring is transparent to applications.

*** Figure 8-9 A Mirrored Disk Volume ***
*** Introduction to Tandem NonStop Systems, S8031-053, part number 82503 ***

Similarly, fault tolerance for the transfer of data between I/O devices and applications is achieved through dual I/O channels, dual ported device controllers, and dual ported disks.

Also, the NonStop architecture contains a number of features designed to minimize the affect of a power failure. Each processor, in addition to having its own power supply, has the ability to draw power from a battery backup for memory. Each device controller has the ability to draw power from dual independent power supplies.

THE OPERATING SYSTEM

The GUARDIAN operating system is a message-based operating system. This differs in concept from the operating systems of other vendors in several ways. First, GUARDIAN is comprised of a collection of programs, each called a process, running in multiple processors. The priority in which the processes are serviced is determined by a priority number assigned to each process. The processes communicate with each other through a common message system, which works as follows:

- A requestor process sends a request (such as a request for an I/O) to another process known as a server process.
- The server processes the request.
- The server sends a reply to the requestor.

Because GUARDIAN is message-based, the Tandem network environment becomes a simple extension of the local operating system. Messages may also be sent to different nodes on the network. EXPAND is the networking software which extends the capabilities to the network.

Major Guardian 90 Software Components

The following table identifies some of the major software components in the Tandem Guardian 90 operating system.

MAJOR COMPONENT	DESCRIPTION
Guardian 90 Operating System Processes	The Guardian 90 operating system processes provide the low level message system, monitoring, and management capabilities. Specific modules include: • Dynamic System Configuration (DSC) • Operator Interface and Remote Maintenance Interface (RMIP/MIOP) • I/O Processes, including DP2 • Interprocessor Bus Monitor (IPBMON) • Memory Manager (MEMMAN) • Monitor Process (MONTOR) • Messenger Process (MSNGER) • IOP processor (IOPROCR)
Guardian 90 Executable Programs	These programs can be run from the Command Interpreter (CI) and provide various utility functions including: ALARMOFF, BUSCMD, COPYDUMP, LIBTRACE, LIGHTS, PASSWORD, RCVDUMP, RELOAD, TAPECOM.
System Utilities	System utilities are provided to configure and maintain the system. Some user actions require privilege and are recorded in the Safeguard audit logs. The system utilities include: Peripheral Utility Program (PUP), File Utility Program (FUP), BACKUP, RESTORE, BACKCOPY, Disk Compression (DCOM), Disk Space Analysis Program (DSAP), DISKGEN, ORSERV, and Remote Console Process (RCP).
Input/Output (I/O) Processes	The I/O processes provide interfaces for the user to access and manipulate peripheral hardware devices. These servers are accessed by users calling the Guardian 90 operating system procedure calls (PROCS) in the file system (e.g. Open, Read, Write). The file system PROCs communicate internally by sending messages to the I/O processes to request the appropriate service on behalf of the user. The I/O processes include the new IOPRM module in the D-series Guardian 90 operating system.
File System	The file system provides a set of callable procedures for the user and system processes to access the message system (and through it, to services provided by other processes).

MAJOR COMPONENT	DESCRIPTION
Event Management System (EMS)	The Event Management System (EMS) is the standard interface for creating system event messages. Each subsystem defines a standard set of tokens in its Data Definition Language (DDL) to report system events. The Subsystem Programmatic Interface (SPI) provides a standard programming tool for interprocess communication.
Transaction Monitoring System (TMF)	The Transaction Monitoring Facility (TMF) runs in privileged mode as an extension of the I/O processes to capture disk data before and after images. TMF provides transaction back-out, roll-forward, and two-phase commit services for user applications and other Guardian 90 subsystems.
EXPAND	EXPAND software drivers for local fiber optic (FOX-ring) system interconnection of up to 255 Guardian 90 systems into a single system image. EXPAND is an extension of the Guardian 90 operating system that creates a processor complex with the same integrity, reliability, and security as a single system.

DEFINING USERS IN THE TANDEM ENVIRONMENT

Local Users

Each individual user in the Tandem environment belongs to a single group. The individual user number and the group number combine to provide a local userid in the form:

groupid,userid

where **groupid** is an integer from 0-255 that identifies the group to which the user belongs, and **userid** is an integer from 0-255 that identifies the user.

A user may also be identified in a user name format,

groupname.username

where **groupname** is the name of the group to which the user belongs, and **username** represents the name of the individual user. The two forms of user identification are interchange-able, and are both defined when a user is added to the system. For instance, a user named AUDIT.JOE might also be represented as numeric userid (40,2).

It should be noted that it is the combination of the groupid and userid which uniquely identify the system user. Thus, while the userid of 2 may correspond to JOE in the group entitled AUDIT, the same userid of 2 may correspond to an entirely different name in another group.

Remote Users

In a networked environment, a distinction must be made between local and remote users. This concept is explained in greater detail in the section entitled "Network Security".

PROCESSES

A process in the Tandem environment is the basic self-contained entity in the computer. Tandem processes can exist as system processes or user processes. The original object code for a program is compiled, and upon initial execution is dynamically bound to produce a file in executable format from which a process can be started.

At any given point in time, only one process is running in a processor. A message system within the operating system facilitates communication between different processes. One process, the requestor, sends a message to another process, known as the server. Each process has its own message queue known as $RECEIVE which is used to read incoming messages.

Messages are continuously sent back and forth between processes throughout the network in a Tandem environment. Messages may be requests to start other processes. In other cases, messages might consist of status notification, [sometimes referred to as "I'm Alive" messages].

TANDEM ADVANCED COMMAND LANGUAGE (TACL)

TACL is the Tandem command interpreter that provides the user with a set of commands or functions that allow the management of system resources, such as files and processes. Generally, there is a software fault tolerant TACL process pair running on each terminal that can process only basic terminal level commands.

When a user logs on, TACL reads the TACLINIT file which helps in the initialization of TACL. TACLBASE is then accessed in a shared read-only extended memory segment. TACLBASE contains all of the TANDEM documented commands needed to use TACL as a command interpreter, such as commands to obtain file information, and commands to gather data on other users and processes. TACLBASE is only reloaded when the operating system installation is performed, associated with SYSGEN and Cold-Load. The INSTALL program replicates TACLBASE into a read-only virtual extended memory segment, which cannot be modified once it is created.

The TACLLOCL data file is read next, and is used by system management to perform local initialization of all users. TACLLOCL can be used to display "No Trespassing" or "Welcome" messages, as required by local laws and customs.

Next TACL reads the individual user's TACLCSTM file, which contains any personal command definitions that the user may have supplied. If a TACLCSTM file does not exist, TACL will automatically create one if the user profile indicates TACLCSTM is allowed. In addition, users can customize their TACL session by creating function key definitions, defining aliases for commands, and defining macros. Once TACLCSTM is invoked, users may then issue commands in their TACL session.

The OLTP Environment

One of the features of the Tandem environment that has contributed to the popularity of the architecture is Online Transaction Processing (OLTP) capability. The software components that comprise the OLTP environment can include:

- PATHWAY Terminal and Transaction Control
- Nonstop SQL Database Access
- Transaction Monitoring Facility (TMF) for Data Integrity

*** See Figure 4-1 Tandem Application Environment, Introduction to Tandem ***
*** Nonstop Systems, S8031-010, part number 82503 ***

PATHWAY terminal and transaction control supports the processing of a large number of transactions concurrently. The Tandem OLTP environment is based on a requestor-server structure, where the requestor program accepts information in the form of a request for information from a terminal and passes it on to a server, which processes the request for information by accessing a database. The Terminal Control Process (TCP) is a Tandem-provided control program which facilitates the development of applications in a multi-terminal environment. Requestor programs are generally written in SCOBOL, a Tandem product similar to standard COBOL. Server programs can be written in any number of languages supported by Tandem, including COBOL85, C, Pascal, FORTRAN, BASIC, or Transaction Application Language (TAL).

The requestor/server structure supports a parallel processing environment, with multiple processors servicing requests simultaneously, and different portions of an application running in multiple CPU's.

PATHWAY

PATHWAY functions as the Tandem environment's online transaction processing facility. Because the PATHWAY environment is based on a requestor-server structure, PATHWAY applications can take advantage of the Tandem multiprocessor architecture. Since a server is capable of running in any processor in the system, different components of an application can be running simultaneously as shown below.

*** Figure 4-6 Effective Use of Multiple Processors, Introduction to Tandem Nonstop Systems **
*** S8031-015, part number 82503***

The elements of a PATHWAY environment are:

Application Programs

<u>SCREEN COBOL (SCOBOL) Programs</u>

SCREEN COBOL programs are the front-end programs that control the display of the screens, accept input data from the terminals, and send data to the back-end server programs for processing via the Terminal Control Process (TCP).

<u>Server Programs</u>

The server programs receive requests from the SCREEN COBOL programs, take the appropriate action, and reply back to the SCREEN COBOL programs. A request may be made to retrieve information from a database, or add/update information to a database. Typically, server programs are developed in COBOL, TAL, or FORTRAN.

Terminal Control Process

The Terminal Control Process (TCP) allows for the operation of multiple terminals in the PATHWAY environment by maintaining separate SCREEN COBOL code and data areas for each terminal defined under its control. In other words, it coordinates communications between the terminals, servers, SCREEN COBOL programs, and PATHMON.

PATHMON

The PATHWAY Monitor (PATHMON) is the central control process which executes PATHCOM commands for PATHWAY system operations. PATHMON accepts requests from the TCP to communicate with a particular server. PATHMON passes the server identification back to the TCP, which can then open the server process.

PATHCOM

PATHCOM is the communications interface used to define and manage objects in the PATHWAY environment. It consists of a set of commands that are passed directly to PATHMON.

Terminals

Terminals are I/O devices used to access and receive information from applications running in the PATHWAY environment. PATHWAY also supports other I/O devices, such as card readers, bar code readers, automated teller machines (ATM's), and point of sale devices.

A PATHWAY application can be distributed across different physical and geographical locations by having different PATHWAY systems communicate across the Tandem network, or by having the components of a PATHWAY distributed across an EXPAND network.

*** Objects Within a PATHWAY System Fig 1-1, S5046-001 ***

ENSCRIBE

A record manager named ENSCRIBE processes a program's I/O requests by transferring records between secondary storage and main memory. While the NonStop SQL database management system processes calls the NonStop SQL database, ENSCRIBE processes requests to and from key sequenced files, entry sequenced files, and relative files, as well as unstructured files, such as EDIT files and object files.

NONSTOP SQL

The database management system in the Tandem NonStop environment is NonStop SQL, an SQL-based relational database language. There is a NonStop SQL programmatic interface which allows programmers to embed SQL statements within application source code, as well as PATHWAY applications.

TRANSACTION MONITORING FACILITY

The Transaction Monitoring Facility is responsible for ensuring overall data integrity, protecting the database from transaction failures, system failures, and hardware failures. TMF can monitor transactions so that if the transaction fails for any reason, TMF has the ability to "back out" all of the updates that were made by the transaction up until the point of failure. The transaction can then be sent back through the system for processing. Programmers must help facilitate this capability by marking the beginning of the activity surrounding a transaction and the end of activity within the applications. TMF manages the transaction backout function using audit records, which are actually before and after images of database records with respect to application of the transaction.

STORAGE MEDIA

A disk file name in a Tandem environment consists of several elements beginning with a volume. A volume name is always preceded by a dollar sign ($) and is further subdivided into subvolumes. Within a subvolume, lies the lowest level of a file name, known as a diskfile. Therefore, in a Tandem environment a file is identified by the volume and subvolume it resides in, as well as its diskfile name.

```
$SYSTEM.SYS21.DATA
```

The above would identify a diskfile named DATA residing on volume $SYSTEM, in subvolume SYS21.

INSPECT DEBUGGING TOOL

INSPECT is an interactive debugging tool which programmers can use to aid in the development and testing process. It allows the programmer additional control over the execution of the program, as well as the ability to interrogate and modify working storage. Due to GUARDIAN's design, INSPECT is not allowed to modify code in execution.

SECURITY IN THE TANDEM ENVIRONMENT

There are some security features that are inherent in the GUARDIAN operating system. The security for many Tandem environments is comprised of these features, along with features built into the applications. SAFEGUARD, Tandem's extended security package, can be used to provide additional controls over Tandem security and to facilitate administrative tasks.

Auditors reviewing controls in SAFEGUARD environments should become familiar with features described in both sections 5 and 6.

SECTION 4
GUARDIAN SECURITY

The security design should use clear naming conventions and resource protection schemes that are consistent with business objectives. Users should be separated into groups based upon job responsibilities. Security may be administered at the group level.

The GUARDIAN operating system provides basic security features. The assessment of extended security features in a GUARDIAN only environment generally will have to be attained through analysis of security features built in at the application level.

GUARDIAN security provides three basic levels of resource protection using security strings. The security strings consist of parameters that identify the access allowed at each level.

Level of Protection	**Protection Provided**
User Level	Protection from unauthorized system access using basic userid/password protection
Diskfile Level	Protection of data files by assigning ownership responsibilities and access levels
Network Level	Control of access to and from remote network nodes.

SYSTEM SECURITY PROTECTION

Basic Diskfile Security

The protection of a single file under GUARDIAN, whether it is a data file or executable file, is established by defining a GUARDIAN security string. In some installations, this may also be referred to as a GUARDIAN security vector. The protection of files in the GUARDIAN environment may be distinguished at two levels: the local environment, which may be considered the immediate node on the EXPAND network, and the EXPAND network environment, which includes not only the local node, but all external nodes which may access the local node via the EXPAND network.

The GUARDIAN security string is a four character positional string associated with each file. It is physically maintained in the diskfile's file label and describes the protection level for the file in the format:

RWEP

where:

R	specifies read access to the file
W	specifies write access to the file
E	specifies who can execute the file (executable files only)
P	specifies who can purge, rename, or compress the file

Valid security string position settings in the local environment are as follows:

LOCAL SECURITY SETTINGS

O	Only the owner of the file on the local system can perform the action
G	Any member in the same group as the owner of the file on the local system may perform the action
A	Any user on the local system may perform the action
-	Only the local super ID may perform the action

Thus, for the file with security string "GOA-" and owner (42,7)

Action	**Authorization to Perform Action**
READ	Anyone on local group 42
WRITE	The local owner, 42,7
EXECUTE	Any local user
PURGE	Local super ID, 255,255

The following positional security string settings define security to EXPAND network users as well as local users:

NETWORK SECURITY SETTINGS

U	Only the owner of the file on the local system or network can perform the action
C	Any member in the same group as the owner on the local system or the network may perform the action
N	Any user on the local system or the network may perform the operation.

Thus, for the file with security string "CONU" and owner (42,7)

Action	**Authorization to Perform Action**
READ	Any network user belonging to group 42
WRITE	The local owner, 42,7
EXECUTE	Any network user
PURGE	Any network user defined with userid 42,7

For the file with security string "NGAC" and owner (42,7)

Action	**Authorization to Perform Action**
READ	Any network user
WRITE	Any member of local group 42
EXECUTE	Any local user
PURGE	Any network user belonging to group 42

GUARDIAN diskfile security is generally established by a security administrator using the File Utility Program's (FUP) "FUP SECURE" command. To secure the file NOTES with the security string "CUCU", the following command would be entered:

```
TACL 2> FUP SECURE NOTES, "CUCU"
```

The security strings may be defined explicitly when a diskfile is defined, or a default protection record may be generated. This topic is described in greater detail in the section entitled "DEFAULT PROTECTION OF OBJECTS".

The GUARDIAN security string for a file can be verified using the FILEINFO command.

```
TACL 1> FILEINFO NOTES
$BOOKS1.LSWORK
```

results in:

```
CODE          EOF        LAST MODIFICATION       OWNER      RWEP
NOTES 101     21484      10-APR-90 15:16:56      147,36     "AOAO"
```

The example illustrates that the file NOTES resides on volume $BOOKS1, subvolume LSWORK, and has a GUARDIAN security setting of "AOAO", indicating that anyone may read or execute the file, but only the owner may write or purge it.

Default Protection of Objects

Under GUARDIAN, when a file is created, the creator is automatically designated as the owner of the file. Each user on the system has a default security string associated with his/her individual userid. This default security string is automatically assigned to the newly created file and may be changed by the individual who establishes the userid, or the user himself/herself. The user has the ability to change the default security setting permanently (with the DEFAULT command), or strictly for the current session (using the VOLUME command).

The GUARDIAN default security settings should be defined with an "OOOO" security string so that only the owner of the file may access the file. This way, the granting of access to other users may be provided only by deliberate action on the part of the owner.

The USERS command may be used to determine a user's default GUARDIAN security setting. The following command lists the default security settings for all super group members:

```
TACL 1>     USERS SUPER.*
```

which results in:

```
GROUP      USER      I.D. #     SECURITY   DEFAULT VOLUMEID

SUPER      .MARY     255,015    OOOO       $SPOOL.MARY
SUPER      .ROBIN    255,200    AAAA       $SPOOL.ROBIN
SUPER      .SPOOL    255,030    AAAA       $SPOOL.SPOOLER
```

In this example, SUPER.ROBIN has a default GUARDIAN security setting of "AAAA", which means that anyone defined to the system may read, write, execute, or purge files created by SUPER.ROBIN which have not been explicitly re-secured. In contrast, SUPER.MARY has a setting of "OOOO", providing only the owner, SUPER.MARY, these capabilities for files that she owns.

Tape Security

The GUARDIAN security strings are maintained on the physical volume on which the diskfile resides. Thus, when a file is backed up to tape, the security string is transferred with the contents of the file itself. As a result, the key areas of concern are the backup of the contents of the diskfile to another media, the restore of the file from that media, and the control of the contents of the file once the file is restored.

Though a key aspect of the security of any tape file will lie in physical controls surrounding the tape itself, the GUARDIAN environment contains features that transfer the controls of the original diskfile to the restored version. The BACKUP program is the primary means of backing up a file to tape. The RESTORE command can be used to recover the backed up file from tape.

When restoring the data, the MYID attribute may be set during the restore to replace the original owner of the file with the userid of the individual restoring the tape. The security settings are then set to the default security setting of the person restoring the file. If the MYID attribute is not set, the file is restored with original security settings intact.

The NOMYID attribute may be specified during BACKUP to prevent use of MYID during RESTORE. If an attempt is made to RESTORE this tape with the MYID option, the RESTORE will abort with an error code.

The NOMYID attribute should be specified whenever a file is backed up to tape. One way in which this can be accomplished is through setting a bind option to disable MYID.

The auditor should review the installation's security policy or documented operations procedures to determine whether the NOMYID attribute is applied.

Process Security

Processes running in the Tandem environment are identified in a manner similar to that of the GUARDIAN userids. Processes are uniquely identified in the form:

processorid,processid

The processorid identifies the specific processor number from 0 to 15 on which the process is running. The processid, an integer from 0 to 255, uniquely identifies the process within the processor. Like userids, processes may also be defined in terms of process names. Process names begin with a dollar sign ($). Within processes, sub-processes, which are preceded by a pound sign (#) might exist.

The concept of fault tolerance extends to processes as well. In a process pair, a primary process executes in one processor, while a backup process runs in another. Through checkpointing designed into the application program, the backup process can always be made aware of the state of the primary process. Thus when the primary process fails, the backup process can resume processing from the point of failure. Critical processes, such as a disk process, PATHWAY, TCP, etc. should run with a backup process as part of a process pair.

The protection of a process involves is determined by the process accessor ids and creator accessor ids for the process. The creator accessor id (CAID) identifies the user who created the process. The process accessor id (PAID) identifies the process itself. The PAID, often identical to the CAID, is the item which is interrogated for the purposes of determining whether the process can access a file.

A list of all processes currently running may be obtained with the following command:

```
TACL 1> STATUS *
```

The runtime object file names should be examined to ensure that critical production processes are running from appropriate production libraries.

Processes Running on a Terminal:

To list the processes currently running on a terminal, enter

```
TACL 1> STATUS *,TERM
```

giving:

```
        > The Processes Currently Running on This Terminal ...

Process          Pri     PFR  %WP   Userid   Program File           Home term

$T712  5,88      150     000        64,18    $SYSTEM.SYS00.TACL     $TM712

$T712  B 9,67    150     001        64,18    $SYSTEM.SYS00.TACL     $TM712
```

The above example illustrates that the process name is $T712, running in CPU 5 as process number 88. It has a backup process running on CPU 9, process number 67. The priority of the process is indicated in the column "Pri". The "R" in the "PFR" column indicates that the process is ready, and not waiting for I/O. A "P" in this column indicates privileged code, while an "F" would indicate that the process is being serviced for a virtual memory page fault.

Further examination would indicate that the process $T712 is running program $SYSTEM.SYS00.TACL under GUARDIAN userid 64,18. The Hometerm column indicates the terminal to which the process is assigned (in this case, $TM712).

Processes Running Under a Userid:

Processes running under a particular userid in the system such as (SUPER.MARY) may be listed with the following:

```
TACL 1> STATUS *, USER SUPER.MARY
```

Programs Being Run by Processes:

Processes running a particular program in the system, such as programs named EDIT within the $SYSTEM volume may be listed with the following:

```
TACL 1> STATUS *,PROG $SYSTEM.*.EDIT
```

PATHWAY Controls

PATHWAY functions as the Tandem environment's online transaction processing facility. It is a set of tools which allow installations to develop online transaction processing applications.

To determine what PATHWAYS are running, enter the following command:

```
TACL 1> STATUS *, PROG $*.*.PATHMON
```

Controlling PATHWAY Configuration

A PATHWAY is defined and configured using a PATHCOM command called SET PATHWAY. A significant attribute is the SECURITY attribute, which defines the users who have the ability to modify the PATHWAY using the SET PATHWAY command. Valid SET PATHWAY SECURITY settings are:

A	Any local user
G	Local group member or owner

O	Owner only
-	Local super ID
N	Any local or remote user
C	Any member of the owner's community (a local or remote user with the same group ID as the owner)
U	Any member of the owner's user class (a local or remote user with the same group ID and userid as the owner)

To determine the users who can modify the configuration for a given PATHWAY, Enter

```
TACL 1> PATHCOM $pathwayname
```

to get into PATHCOM. Then the command

```
=INFO PATHWAY, OBEYFORM
```

can be run so that the field labeled "SECURITY" can be examined. This security vector identifies who can modify the PATHWAY and can take on the same values as a RWEP security string for a diskfile.

Controlling the Execution of PATHWAY Applications

Execution of an application program in a PATHWAY environment can be initiated from a PATHCOM defined command terminal using a RUN PROGRAM command.

The SET PROGRAM command, in the format:

```
SET PROGRAM program-attribute
```

is used to establish the program attributes within a PATHWAY application. It is executed with the appropriate parameter to indicate the program attribute that is being set. SET PROGRAM ERROR-ABORT, for example, can be used to specify action to be taken when a program error occurs.

SET PROGRAM OWNER is used to define the program owner which is allowed to issue the RUN PROGRAM command. Similarly, SET PROGRAM SECURITY is used to specify other PATHCOM users who can issue the RUN PROGRAM command for that program. SET PROGRAM SECURITY may be set to the same attributes as SET PATHWAY SECURITY.

To identify the security vector set in the SET PROGRAM command, the command

```
=INFO PROGRAM program-name, OBEYFORM
```

can be entered in order to determine the users who can modify program attributes.

Clearing Deleted Files

When a file is deleted, the block locations on the disk are marked as available for reuse by other files. The contents of the block locations are by default not erased or overwritten because of the performance penalty implicit in overwriting files; therefore, the original information may be available to privileged programs which access the disk directly.

The CLEARONPURGE option, available for diskfiles, may be set to physically overwrite the residual data from any files which have been deleted (i.e. purged). Because every block of the purged file is overwritten with binary zeros, CLEARONPURGE requires extra I/O operations and disk channel time. This added overhead increases proportionally with file size. The increase is hardly noticeable on a small file, but can be many minutes of elapsed time on a large file. Most commercial sites make little or no use of CLEARONPURGE because of the added processing overhead.

CLEARONPURGE should be designated for files containing particularly sensitive data. Other files should be evaluated by the organization to weigh the risk of not clearing the deleted file against system performance considerations.

Under GUARDIAN, CLEARONPURGE may be checked for a given diskfile by issuing the following command:

```
TACL 1> FUP INFO filename, DETAIL
```

The auditor should check to determine whether CLEARONPURGE is set for any critical files.

Detection of Orphan Files

When employees have been terminated or have transferred to other departments, procedures should be in place to ensure that their userids are removed from the system, and that any files that they have created or own are transferred to other users or deleted. In network environments, the controls must extend to all possible nodes for which the user had access capabilities.

An orphan file is a file which is owned by a nonexistent user. In most cases, the user has left or transferred from the organization and the userid has been deleted, but the files owned by him or her still exist. The security risk involved lies in the possibility that the same userid could be assigned to a new user in the future. The new user would then inherit ownership of all the orphan files.

Procedures should be established so that when users no longer require system access (resigned, terminated, transferred, etc.) ownership of existing files is appropriately transferred in both the local and network environments or the files are deleted. Some system administrators reserve several userids to intentionally create in-accessible orphan files, which they use as temporary holding places for files that are in the process of changing ownership.

The Disk Space Analysis Program (DSAP) may be used to detect files that a userid owns. DSAP should be run against a userid prior to assigning that userid to an individual. This would display any orphan files that might still exist under that userid. The format of the DSAP command is:

```
        TACL 1> DSAP $volumename, USER userid, DETAIL
```

For example,

```
        TACL 3> DSAP $SYSTEM, USER 254,10, DETAIL
```

searches volume $SYSTEM for files belonging to user 254,10, resulting in:

```
        PAGE 0    DSAP -- $SYSTEM on \NY -- ?????????.???????? --

        Disc Space Analysis Program -- T9074Xnn - (ddMMMyy)

        Summary of space use for ?????????.????????? on $SYSTEM

        No files allocated.
```

The example reveals that the user, 254,10 no longer exists on the system, and that no files with 254,10 listed as the owner reside on the volume $SYSTEM.

The auditor should confirm that the installation has procedures for checking for orphaned files when userids are deleted, and when new userids are assigned.

General Application Security Controls

In addition to the physical security controls supplied by the installation and the access and authentication controls provided by GUARDIAN (and SAFEGUARD when implemented), important security controls can be defined within each individual application. Applications may add an additional level of user authentication. Messages and diagnostics may be produced to supplement the monitoring function.

Considerations for Controls Built Into Applications

In general, security controls which are built into individual applications are difficult to monitor and audit. These controls provide an additional level of administration that must be addressed on top of GUARDIAN security controls. As a result, the addition of application level security should be reviewed closely by the applications development group, along with operations, technical support, security administration, and audit to determine whether the benefits that the additional security provide would be worthwhile, given the additional maintenance and administrative overhead that can result.

Design Guidelines

Applications which do require their own built-in security should be consistent with the organization's security policy and take into account the following issues. The list provided is by no means a complete list of considerations of application security, but highlights key points.

- An adequate organizational change management process should be established, with thorough review and testing of application code by all parties affected prior to implementation.
- File names produced by the application should be consistent with the organization's naming conventions, so that elements such as data files, programs, etc. can be easily identified.
- The application should be appropriately defined to a monitoring facility so that operations can closely watch the events associated with the application.
- Operator intervention required to run programs should be minimized.
- If the application performs its own authentication, access attempts should be appropriately logged. These audit records should be available for review by appropriate personnel (operations, audit, security administrator, etc.).
- In some cases, the applications might need to log certain events to facilitate security monitoring. See the "Pathway Considerations" section.

SPECIAL PRIVILEGES

User Classification

The grouping of all system users into an appropriate classification is especially significant in the Tandem environment, since certain user privileges are automatically associated with different classifications of users.

There several classifications of userids in the TANDEM environment.

Super ID (255,255):	The most powerful userid defined on the Tandem system. This user has the ability to access any file, process, or device on the local system without restrictions.
Super Groups (255,*):	Also called "system-operator id's", these are used on operations-related responsibilities such as management of system files, system configuration, and the abilities to start and stop various hardware devices. Super group members are also able to execute certain privileged commands.
Group Managers (*,255):	Are responsible for one or more general users which comprise a group. A group manager can add/delete users from the group, can sign on as any group member without the user's password and has access to any files owned by the group.
General Users:	Users at the application level.
NULL.NULL (0,0):	This user exists when the system is first initialized, and should be deleted.

The use of group naming conventions should easily identify an individual's job responsibility area. Commonly, groups might coincide with project or department teams, such as audit, accounting, purchasing, etc. Groups also might be defined by job function, such as Operations, Security Administration, Programming, etc.

A list of users in the each group can be obtained through the USERS command from TACL, and providing the group name or group number. Thus, a list of the members defined to the super group can be obtained with following command:

```
TACL 1> USERS 255,*
```

The list should be examined to confirm that the users listed are appropriate.

A list of users can also be obtained by user name, such as

```
TACL 1> USERS SUPER.*
```

Consideration should be given to avoiding the assignment of group manager ids. Groups manager ids by default, have special privileges associated with them such as the ability to sign on as any user within the group without the use of that user's password. Instead, managers can be defined as group members, and necessary privileges can be explicitly assigned.

However, listing by userid is strongly recommended, since there is little significance in the actual name of "SUPER". Installations can modify the names of any users, but the number 255 is always associated with a super group or super user.

User Authentication

In addition to assigning each user a userid so that he or she can be uniquely identified, the user should be authenticated to ensure that a different individual is not merely posing as that user to gain unauthorized access to the system. This can be accomplished in a variety of ways, but the most widely-used method of user authentication remains the use of passwords.

In the Tandem environment, the authentication of a user at logon time can be verified against those users defined to GUARDIAN by issuing a call from a program to a procedure called VERIFYUSER. VERIFYUSER, in this case, would either be called by the TACL program at logon time, or by an application invoked directly. In some environments, GUARDIAN user authentication at logon time may be bypassed in favor of an alternative user authentication scheme built into a front end application.

For those installation-developed applications which perform their own VERIFYUSER calls, the code should be carefully reviewed to determine whether VERIFYUSER is implemented as intended.

Naming Conventions

Accountability is a key component of effective security control in any environment. Each user who accesses a system should be uniquely identified, and controls should be in place to allow for the monitoring of high risk activity that takes place during his/her session.

The security design should use clear naming conventions and resource protection schemes that are consistent with business objectives. Users should be separated into groups based upon job responsibilities and security may be administered at the group level.

Thus, a groupname might coincide with a business unit, such as AUDIT, or SALES. Logon with a username should be required, so that the chances are greatly reduced that unauthorized users can logon to the system by choosing userids and trying common passwords on a trial and error basis. This will control the intruder who gains access to the system by attempting to use all userids from 001,001 thru 255,255 until successful access is obtained.

Setting the TACL flag NAMELOGON will require logons with names only. The organization's TACL programs run at logon time should be reviewed to determine whether users are restricted from logging on via userid. The auditor may test this feature by attempting a logon or by reviewing the TACL for a nonzero NAMELOGON parameter.

The Super ID

The super ID (255,255) is the most powerful userid available in the TANDEM environment. Proper management of the super ID is critical to the integrity of the Tandem environment, since the super ID has the ability to bypass security controls that are established for general users.

Abilities of the Super ID

Use of the super ID should be restricted to emergency situations, since it has the ability to bypass normal security controls in the local environment. Unless adequately controlled, the super ID has the following abilities:

- It can logon as another userid
- It can read, write, or purge any local file
- It can bring up or take down any device
- It can modify privileged code

Tasks Which Require Super ID Capability

In short, the super ID can perform any operation on the system. As a result, the super ID should not be used for day-to-day operations. Most ongoing tasks can be performed without the super ID. PROGID programs are just one way to provide special privileges without using the super ID. The tasks which DO require the super ID are:

- Licensing and revoking the license of a program
- Setting PROGID for programs that need to run under the super ID
- Initializing Nonstop SQL
- Add/Delete User groups

Operations Tasks Which Do NOT Require Super ID Capability

Many operations tasks such as those listed below can be performed without using the super ID.

Spooler Control	User should be logged on as a member of the super group and have EXECUTE access to the SPOOLCOM program.

Device Control	User should be logged on as a member of the super group and have EXECUTE access to the PUP program to activate or deactivate devices.
System Backup	Give the super group EXECUTE access to a PROGID copy of the BACKUP program with PROGID set to the super ID.
TMF Control	User should be logged on as a member of the super group and have EXECUTE access to the TMFCOM program to start or stop TMF.
System Time Control	Setting the system time requires the user to log on as a member of the super group.

The status of the super ID may be verified through GUARDIAN:

```
TACL 1>USERS 255,255
```

The auditor should identify whether the Super ID is active and whether its application is appropriate.

Authorization to Define Security to Objects

In the GUARDIAN environment, supergroup managers and the super ID have the ability to define security strings for objects. The ability to add users to the system in GUARDIAN is provided by default to group managers. A group manager with userid (50,255) has the ability to add or delete users from their own group, in this case group 50. This is accomplished using the ADDUSER and DELUSER commands.

A review of users who have access to the ADDUSER and DELUSER programs can be obtained by the following commands:

```
TACL 1> FILEINFO $SYSTEM.SYSTEM.ADDUSER

TACL 2> FILEINFO $SYSTEM.SYSTEM.DELUSER
```

The installation may choose to restrict access to these programs to group managers who by default have the ability to define users to the system. By doing so, the installation confines the ability to add users to the super ID.

RESTRICTED FUNCTIONS

Licensing

In the Tandem environment, the term "licensed" refers to any program that contains privileged operations, or operations that are reserved for use by the operating system. Programs running in privileged mode can execute privileged code, and access operating system tables. Using the capability of licensing, the installation can allow a user other than the super ID to execute privileged code.

<u>Capabilities of Licensed Programs</u>

Licensed programs can bypass security controls, disrupt the network, and, in short, do almost anything that the super ID can do. These programs present special security risks, since they can also:

- Modify protected memory containing instructions and data

- Change an intruder's PAID to gain the privileges of other users, and subsequently change files.

- Execute privileged instructions.

- Access system global data space.

- Manipulate physical hardware resources (e.g., stop a processor).

Only the super ID can license a program. This is accomplished by setting a LICENSE bit in the object code file (Code 100 file).

<u>Guidelines for User Written Licensed Programs</u>

In general, no user programs should be licensed, since licensed programs have the ability to bypass security checks. If it is determined that user programs must be licensed, formal approval procedures should be set up to provide proper review of the need to license the program. The approval process should include:

- Documentation of the program's purpose and the reason for requiring that it be licensed.
- Management approval
- Review of Source Code
- Does the source code recognize specific users for certain capabilities?

- Does the source code appropriately modify operating system control blocks?
- Does the code change the PAID?
- Does it create processes on behalf of other users?
- Does it contain unauthorized, non-documented functions?

Also, in order to properly segregate duties, licensed user code should be compiled and bound by someone other than the original programmer.

Control of Licensed Programs

Licensed programs should be monitored closely to ensure that their activity is consistent with the organization's security policy and guidelines. This might include restricting the number of individuals who can execute them.

Because the code in licensed programs can be closely coupled with the features of specific GUARDIAN releases, all licensed programs should be reviewed when new software releases are implemented to ensure that the code is not adversely affected by the new release.

To obtain a list of all licensed files for a volume,

```
TACL 10> DSAP volumename, LICENSED
```

The command:

```
TACL 1> DSAP $SYSTEM,LICENSED
```

results in:

```
User Name/ID          Filename                Type Code ...

SUPER.SYS
(255,0)               SYS00.ADDUSER           100L
                      SYS00.BACKUP            100L
                      SYS00.CMP               100L
                      SYS00.CMPLIB            100L
                      SYS00.DEFAULT           100L
                      SYS00.DELUSER           100L
                      SYS00.DSAP              100L
                      SYS00.FUP               100L
                      SYS00.PASSWORD          100L
                            .
                            .
                            .
```

Where the "L" in the type code column denotes that the program is licensed.

To determine who can execute licensed programs, a FILEINFO (GUARDIAN) command can be executed against the licensed filename.
The auditor should inquire whether any user programs are licensed and check for appropriate documentation of business justification, and supervisory review/approval.

System Files

Tandem supplies various utilities in system files residing in the $SYSTEM.SYSTEM and $SYSTEM.SYSnn subvolumes. Examples of these files include PASSWORD, TEDIT, TACL, PUP, etc.

Ownership of System Files

System files distributed by Tandem are by default owned by userid 255,0. If the userid 255,0 has not been added to the system, the system files that it owns by default should have their ownership transferred to the appropriate individuals.

Critical Processes

All system elements should be secured based on Least Necessary Privilege. The protection of the following critical processes and utilities has special security implications and should be scrutinized more closely:

$0	EMS Primary Collector
$Z0	EMS Compatibility Distributor
$CMON	Command Monitor
$IMON	INSPECT MONITOR
$DMnn	DEBUG Monitors
$ZSMP	SAFEGUARD Monitor

Critical Utilities

If licensed, the following tools also have special security implications:

DIVER	Brings a CPU Down
TANDUMP	Raw Edit a Disk
CMI	Can Trace a Terminal
MD2	Read/Write System Memory
MIO	Change I/O Device Configuration
SNOOP	Examines TMF active transaction table from memory, and audit trails on disk

Note: These programs are created for Tandem internal development use and are not released on customer SUT distribution tapes. They are listed here as a precaution in case a copy is inadvertently been installed on a customer system.

Access to System Files

Access to system files supplied by Tandem should be adequately restricted. Write or purge access should not be allowed to system files. Security settings for the system files listed above can be verified using the FILEINFO command.

FILEINFO should also be run against $SYSTEM.SYS*.* to obtain a list of all files in the $SYSTEM.SYSTEM and $SYSTEM.SYSnn subvolumes so that they may be reviewed.

The version proc timestamps for system files should be compared against those provided by Tandem when the operating system was last updated.

```
TACL 2>VPROC $SYSTEM.SYSTEM.*
TACL 3>VPROC $SYSTEM.SYSnn.*
```

can be used to determine the time date stamps of the system files currently being used in production. Third party entries in these subvolumes should be checked to ensure that VPROCs are present.

The contents of the $SYSTEM.SYSTEM and $SYSTEM.SYSnn subvolumes should be examined for duplicate entries. A duplicate of a program residing in $SYSTEM.SYS01 could be placed in $SYSTEM.SYSTEM and thereby supersede the original version, since the SYSTEM subvolume is searched first by the system. (Using SAFEGUARD, create access should be denied to all unauthorized users for these subvolumes. See Section 6.)

EXTENSIONS AND MODIFICATIONS

$CMON Generalized User Exit

Some installations (particularly those without SAFEGUARD security) may use the $CMON generalized user exit from TACL to implement logon controls.

Function of $CMON

The requests might include logon commands or RUN commands. In other words, the $CMON process is a user-written program and typically is used to:

- Prevent certain users from logging on during certain time periods.
- Prevent certain users from logging on at certain terminal locations.
- Control of the priority of process.

Commands Recognized by $CMON

The $CMON process monitors and controls requests to the command interpreter. The command interpreter (TACL) acts as a requestor, and $CMON as a server in a requestor/server relationship. Each time one of the following commands is given,

ADDUSER
ALTPRI
DELUSER
LOGOFF
LOGON
PASSWORD
REMOTEPASSWORD
RUN (Implicit or Explicit)

the command interpreter sends a message to the $CMON process' $RECEIVE file. This provides $CMON an opportunity to invoke additional security controls that the installation has built into it.

Control of $CMON Code

Code for $CMON should be well-reviewed and documented prior to implementation. READ and WRITE access to $CMON object code should be restricted to the owner and appropriate individuals who are responsible for maintaining and migrating the code into a production environment.

$CMON documentation should be closely evaluated to ensure that the code is consistent with the organization's programming standards and guidelines. Special privileges for specific userids should not be hard coded into the $CMON source code. Code that alters the priority of programs or alters user passwords should be examined for appropriateness.

$CMON and TACL

When TACL is used in the logon process, access to $CMON should be required. Otherwise, $CMON can be bypassed in lieu of an application. $CMON can also be bypassed by a process if the process has been waiting for $CMON for an extended period of time to run.

```
TACL 1> PPD $CMON
```

may be run to determine whether $CMON is running, resulting in:

```
Name        Primary        Backup        Ancestor
$CMON       7,27           0,26          5,21
```

Reviewing $CMON Code

The location of $CMON object code and the runtime library may be obtained from the following:

```
TACL 2> STATUS $CMON, DETAIL
```

The STATUS $CMON command yields the following:

```
System:  \SECURE                               October 24, 1990  10:48
Pid: 0,26     ($CMON)     Backup
PRIV
Priority: 165
Wait State: %001   (LREQ)
Userid: 255,255   (SUPER.SUPER)
Myterm: $OPR1
Program File Name: $SYSTEM.SYSTEM.CMON     <======== OBJECT CODE
Swap File Name: $SYSTEM.#0122
Library File Name: $SYSTEM.SYSYEM.CMONLIB  <======== RUNTIME
                                                    LIBRARY
Process Time: 0:00:00.006
Process States: RUNNABLE
GMOMJOBID:

System: \SECURE                                October 24, 1990  10:48
Pid: 7,27    ($CMON)          Primary
PRIV
Priority: 165
Wait State:  %001    (LREQ)
Userid: 255,255    (SUPER.SUPER)
Myterm: $OPR1
Program File Name: $SYSTEM.SYSTEM.CMON
Swap File Name: $SYSTEM.#0121
Library File Name: $SYSTEM.SYSTEM.CMONLIB
Process Time: 0:14:34:9666
Process States: RUNNABLE
```

Security surrounding the runtime library should be reviewed to ensure that the $CMON routine is not subject to unauthorized access. The "Program File Name" field reveals the name of the file containing the original object code. The "Library File Name" field contains the name of the runtime library in which the executable code resides. Access to these libraries should be reviewed to ensure that $CMON is protected against unauthorized modification.

Similarly, determination of from where the object code was compiled can be obtained from the following series of commands:

```
TACL 2>  STATUS  $CMON
```

```
System \SECURE

Process                              Pri PFR %WT Userid  Program file    Hometerm
$CMON  B  0,26        165 P   001 255,255  $SYSTEM.SYSTEM.CMON  □□□□

                                              Swap File Name:  $SYSTEM.#0122
$CMON     7,27  165  P  001 255,255  $SYSTEM.SYSTEM.CMON         $OPR1
                                              Swap File Name:  $SYSTEM.#0121
```

This shows the name of the source for $CMON, $SYSTEM.SYSTEM.CMON. A temporary file may be created to hold the source:

```
TACL 3> CREATE BINDTEMP,30
TACL 4> BIND /OUT BINDTEMP/
```

```
BINDER - OBJECT FILE BINDER - T9621C10 - (18AUG89) SYSTEM \SECURE
Copyright Tandem Computers Incorporated 1982,1983,1984,1985, 1986, 1987
@LIST SOURCE FROM $SYSTEM.SYSTEM.CMON
@EXIT
STOPPED: 10,84
CPU time 0:00:00.541
```

TACL 5> FUP COPY BINDTEMP

```
BINDER - OBJECT FILE BINDER - T9621C10 - (18AUG89) SYSTEM \SECURE
Copyright Tandem Computers Incorporated 1982, 1983, 1984, 1985, 1986, 1987

@LIST SOURCE FROM \SECURE.$SYSTEM.SYSTEM.CMON
SOURCE FILE                 B/P       NAME                DATE
$SYSTEM.CMON.CMONS          P         GETBAGDATA               14APR83
                            P         MAILOPEN                 14APR83
                            P         NONSTOP^SETUP            14APR83



$SYSTEM.CMON.COMPCPU        P         TADD
                            P         XRAYNUMOUT

                            P         SCALEDDIV
                            P         TSUB
                            P         DASCII
$SYSTEM.CMON.CMONS B            #GLOBAL
                            B         #GLOBAL
@exit
```

PROGID Programs

In order for a system user to run a program, thereby creating a process, execute authority must be granted to the program object code of the process. In addition, the user must have proper access defined to the files that the process will access. Program file owner adoption (PROGID) programs allow users to gain access to another user's data without explicitly defining the user in a diskfile's access control list. This ability allows the owner of a process to temporarily delegate a subset of his or her abilities without granting the full capabilities of the owner. For example, a super user can create a program to be used for volume backups. By allowing operations personnel to execute this program with the PROGID attribute, super user capabilities do not have to be assigned to the operator.

Security Implications

There are important security implications associated with the ability to PROGID. Each process in a Tandem environment is associated with a Process Accessor id (PAID) and a Creator Access id (CAID). When a process runs, the CAID identifies the user who created the process. The PAID uniquely identifies the accessor of the process, and is checked to determine if file access should be allowed.

When a PROGID program is executed, the PAID is set to assume the privileges of the program owner, regardless of who is executing the program.

Only the original version of the program which was PROGID'ed retains the PROGID function. In other words, if the program is copied or restored from backup, the PROGID capabilities are not carried forward. The owner of the program must explicitly re-enable the program as a PROGID program using FUP SECURE, SETMODE, or SETMODENOWAIT.

Control of PROGID

A formal review process should be established for reviewing a PROGID program chain to ensure that it is coded properly. PROGID programs with bugs in it could unintentionally assign privileges to another user that were unintended.

Careful review should be conducted of PROGID programs which themselves execute other processes. By doing so, accesses made by the other processes are logged to the owner of the PROGID program. In other words, the CAID of the second program is replaced with the PAID of the first, and accountability of the individual running the program is lost.

System programs should not be enabled as PROGID programs.
Furthermore, programs should never be constructed to enter DEBUG or INSPECT mode. When running in DEBUG, an online interactive debugging tool, "patching" the program data can make it possible to defeat whatever security is built into the program.

To list the programs that reside on a volume with PROGID capabilities,

```
TACL 1>DSAP volumename, PROGID, DETAIL
```

This will show the names of all PROGID programs residing on a volume, and by whom they are owned.

For example, the instruction

```
TACL 1>DSAP $SYSTEM, PROGID, DETAIL
```

yields the following result for the $SYSTEM volume:

```
Name/ID               Filename                 Type Code ...

LIBRARY.USER          SYSTEM.LIBRARY           100I
(5,1)

ADMIN.SUE             SYSTEM.CRS               100I
(149,60)              SYSTEM.TPS               100I
```

The "I" next to the type code 100 indicates that the file is a PROGID program.

To determine who can execute PROGID programs, the FILEINFO or INFO DISKFILE commands may be executed against the file containing the executable code.

INSPECT and DEBUG

INSPECT is a debugging tool which allows for interactive control of program execution. The user can execute a program on an instruction by instruction basis while displaying and modifying symbolic data fields, registers, and the current data segment ID. INSPECT can be invoked by execution of the TACL RUN INSPECT command.

Similarly, the DEBUG facility provides a way to interactively debugging a program. It differs from INSPECT in that DEBUG is a low-level debugger that does not allow for symbolic referencing of data fields. For example, while using INSPECT, a programmer could reference the contents of a field named CURRENT-SALARY by using the field name. Under DEBUG, the field contents could only be displayed by referencing the address of CURRENT-SALARY.

PROGID programs should never be constructed to enter DEBUG or INSPECT mode. Similarly, programs running in production should not enter INSPECT or DEBUG mode, since it is possible for an individual to temporarily suspend execution of the program and modify data. An application should call ABEND in case of an application error.

The auditor should review the installation's security policy to see that use of INSPECT and DEBUG are restricted to program development, and not used in the production environment.

The auditor should examine the security settings for INSPECT with:

```
TACL 1>FILEINFO $*.*.INSPECT
```

SYSTEM CUSTOMIZATION

Initializing the Security Environment

Configuring the environment involves defining all of the hardware and software components that make up the system. This can include devices such as disks, tape drives, terminals, and printers, data communication lines, other systems or terminals, and system software, such as PATHWAY.

The INSTALL program is used by operations to generate and install the system after it is configured. INSTALL executes a number of programs that guide the installer through the process, including a program called SYSGEN. SYSGEN is responsible for generating an operating system from the defined hardware and software configuration.

GUARDIAN security runs as part of the GUARDIAN operating system. When GUARDIAN security is "down", then the operating system is "down" as well. As a result, there is no explicit action which is required to activate GUARDIAN security.

SYSGEN Parameters

System generation involves defining, updating, and installing the configuration of the system. The system configuration includes the definition of system hardware (disks, tape drives, printers, asynchronous terminals, etc.) and system software components in a configuration file.

The INSTALL Process

Once the system hardware has been configured, the person performing the system generation process runs an interactive program called INSTALL. This guides the generation process through a number of programs, including installing the operating system and restoring the system image on the system disk.

Some system components, such as devices attached to communication lines, system software such as TMF, PATHWAY, and the Spooler are not defined until a process known as online configuration has been run. Online configuration does not take place until the operating system is up and running.

When running INSTALL, one of the programs executed is the SYSGEN program, which generates an operating system for a given hardware and software configuration.

Location of SYSGEN Parameters

The system hardware configuration that is defined to the SYSGEN process is kept in a file called the CONFTEXT file. The CONFTEXT file contains all of the hardware descriptions applicable to the installation. The CONFAUX file is an auxiliary configuration file that contains the definitions for the INSTALL descriptors which are in the CONFTEXT configuration file.

The CONFTEXT and CONFAUX are used as input to SYSGEN, which generates an operating system in the form of a system image tape (SIT). The SIT is then loaded into the system volume and subvolume, and the system is cold-loaded.

The contents of the CONFTEXT and CONFAUX files should be examined for appropriateness. Information about the files can be obtained with the following command:

```
TACL 1> FILEINFO $SYSTEM.*.CONF*
```

A listing of the contents of the CONFTEXT file can be obtained with the following command:

```
TACL 2> FUP COPY $SYSTEM.SYS00.CONFTEXT
```

Super ID Control and SYSGEN Parameters

A key SYSGEN parameter with audit implications is the ability to provide the super ID the privilege of bypassing access denials. The overriding of denials to the super ID should be strongly discouraged. The super ID should not be used for day-to-day operations, and should not be allowed to bypass any security controls that have been set for it. The ALLPROCESSORS PARAGRAPH of the configuration file should be examined to determine whether the super ID is undeniable. This is determined by the presence of

```
SUPER_SUPER_IS_UNDENIABLE
```

indicating that all denials to the Super ID will be honored.

Controlling the Logon Process

TACL and the Logon Process

There are numerous ways in a Tandem environment to control the logon process. In many cases, logon and user authentication will be controlled by an installation-defined application. (In other cases, the installation will take advantage of SAFEGUARD controls to perform authentication.) The logon process can be controlled by a combination of TACL and $CMON code.

The Tandem Advanced Command Language (TACL) provides an interface between the system user and programs. TACL can call other programs and applications, and can also be used to manage documents and electronic mail, as well as compile/debug programs and perform system management. In some installations (especially with those without SAFEGUARD) it is possible that the user is running a TACL session immediately on logon.

TACL is by default responsible for the logon process in a Tandem environment. Installations may customize the logon process to check for failed attempts, log actions, etc. under TACL by invoking $CMON upon logon. $CMON can contain installation defined routines to check for certain features during the logon process, and can reject the logon.

With TACL, a user can type in the password immediately following the userid when logging on:

```
TACL 1> LOGON AUDIT.JOE, PASWRD4U
```

If the return key is hit after typing the userid, TACL prompts the user for the password without echoing the text. This feature is known as a blind logon and can be stopped with the BLINDLOGON option:

```
TACL 1> LOGON AUDIT.JOE
Password:
```

The blind logon feature should be strongly encouraged in the organization's security policy when TACL is used to control the logon process. Since Tandem terminals allow backward scrolling on screens to reveal previous commands typed in TACL, failure to use the blind logon feature could in certain cases allow anyone to view the password used simply by scrolling the terminal, provided the information is still in the buffer.

Once the password has been entered by the user, TACL makes a call to the GUARDIAN procedure VERIFYUSER to authenticate the user. Userids, logon names, encrypted local and remote passwords, and default "RWEP" diskfile security are maintained on the file named $SYSTEM.SYSTEM.USERID.

Tailoring the Logon Process

Each environment can tailor the TACL logon process by executing various TACL macros on signing on to the system. Initially, these macros are loaded by a file called TACLLOCL. These macros are executed from a file called a TACL Custom File (TACLCSTM) contained in each user's default subvolume. Commonly, macros executed by the TACLCSTM file can give control to a specific application immediately on completion of the logon process.

Invalid Logon Attempts

TACL will not accept logons after three consecutive invalid attempts have been made. After 60 seconds, the terminal will again accept logon attempts. In some installations, it may be possible to freeze ids after a predetermined number of invalid attempts have been made through macros or adding code to $CMON.

TACL and NULL.NULL

When a user logs off, the TACL process is still running, but as a NULL.NULL (0,0) user. As a result, the NULL.NULL user should not be defined as a GUARDIAN userid. When a user logs off, his or her TACL session still runs under user (0,0). If the NULL.NULL user were defined, an intruder could log on as NULL.NULL and disrupt all TACL's running under the NULL.NULL userid by changing their priorities or stopping them.

The USERS 0,0 command could be run to confirm whether the NULL.NULL userid is undefined to the system.

Recommendations for Logon Controls

If the logon process is controlled by an application, or by TACL (and $CMON), the controls used should be clearly documented. Protection of critical files and code used to define the logon process should be adequately restricted.

The $SYSTEM.SYSTEM.USERID file containing all passwords should be appropriately restricted against unauthorized use. The recommended security is "----" or "OOOO" and owned by "255,255". Passwords should be encrypted so that anyone browsing the file cannot determine the system passwords. Similarly, the TACLCSTM "0000" and TACLLOCL "A000" files should be adequately secured so that they cannot be updated by unauthorized users.

The TACL FILEINFO command can be run against $SYSTEM.SYSTEM. USERID in order to determine the security setting for the userid file.

The security for the TACLCSTM or TACLLOCL file can be verified by issuing a FILEINFO TACLCSTM or FILEINFO TACLLOCL command against the user's default subvolume:

```
1> FILEINFO $*.*.TACLCSTM
```

```
$ADM.ROBI
N

             CODE   EOF    LAST MODIFICATION    OWNER    RWEP
TACLCSTM     101    2896   27-APR-90 9:57:36    147,36   "OOOO"
```

Similarly, a FILEINFO $*.*.TACLLOCL may be run to review the security for the TACLLOCL file.

The above example illustrates that the TACLCSTM file on the default subvolume $ADM.ROBIN has security set to "OOOO", allowing only the owner to access the file. The write field should be closely reviewed to ensure that no other users have write access to another's TACLCSTM.

Auditing Attempts to Access System Resources

In the Tandem environment, the term "audit" in the context of a data file refers to the logging of accesses to that file. GUARDIAN itself provides very limited audit features. Generally, if events are audited, the audit capabilities must be built by the user into either $CMON or the applications.

There are no security-related logging features inherent in GUARDIAN. Any logging which takes place in the non-SAFEGUARD environment is implemented in the application.

Database Audit Trail

The Transaction Monitoring Facility (TMF) allows for the maintenance of database audit trails in the form of before and after images of data base update records. TMF manages the transactions that perform functions such as updating a database. TMF should be running at all times; if TMF fails, all applications that are running at the time under TMF transaction management will fail as well.

The primary function of TMF is to ensure the proper completion of all transactions. In the event that the transaction fails, then TMF must ensure that all interim database updates are appropriately backed out. This is referred to as the <u>autorollback</u>, or <u>backout</u> feature in the Tandem environment. TMF will be stopped by the system if the number of audit records exceeds the allocated file size.

Depending upon the criticality of the information stored on the database, the installation may choose to use audit capability on database files containing high dollar transactions. Since there could be considerable overhead in doing so, the determination should be made based upon the importance of the data as determined by an appropriate risk assessment.

The auditor should review the organization's analysis of the criticality of the information maintained on the database in order to agree it with the organization's position on analyzing TMF records.

Audit trails should be backed up to tape regularly to prevent overflows. Tapes should be sent offsite for safekeeping.

To determine which files are TMF protected,

```
        TACL 1> DSAP $*, AUDITED
```

Auditing System Console Activity

Console messages, which include both system-generated and user-generated messages, can be written to a hard copy device by using the PUP CONSOLE command. The PUP CONSOLE command has the ability to disable/enable logging to the hard copy device, as well as to define a new device. In addition, PUP CONSOLE may be used to selectively disable specific messages.

Logging of console activity can also appear in a user-written application process which is named $AOPR, or a disk log file, usually called the OPRLOG.

Activity surrounding the systems console should be reviewed on a regular basis by an appropriate supervisory person. Special attention should be paid to activity that takes place during events such as system failure and recovery, special job runs, etc.

The logging status of operator console messages can be determined using a PUP LISTDEV $0 command. This command gives the status of logging to the disk log file under the STATE header and the logical device number in octal representation of the hard copy device, if any, under the header labeled PCU.

Password Management Features

GUARDIAN does not enforce edit rules for user passwords. (Extended password management controls can be obtained either through SAFEGUARD or by building them into the applications.) However, some basic password management functions are available.

Under GUARDIAN, PROMPTPASSWORD and BLINDPASSWORD can be set to request a system prompt for the password and to suppress echoing of the password on the terminal as it is typed. In addition, GUARDIAN allows for the specification of a minimum password length MINPASSWORDLEN and the encryption of stored passwords (ENCRYPTPASSWORD).

GUARDIAN options should be set to require a password prompt, and to provide for a blind password. A blind password is a password which is not visible on the screen. In addition, the minimum password length should be set at an appropriate level to discourage the guessing of passwords.

Encryption of Passwords

Passwords in the Tandem environment can be encrypted so that someone with access to the password file is still unable to read the passwords. ENCRYPTPASSWORD is the GUARDIAN password program parameter which sets the encryption of passwords.

Passwords should always be set so that they are encrypted (whether they are under the control of GUARDIAN or SAFEGUARD) so that anyone reading the password file cannot read the passwords of the system users. If encryption is not set until after the system has been running for an extensive period of time and there are userids residing on the system that are not encrypted, the system administrator should require all users to modify their passwords so that they are stored in encrypted format.

A determination can be made as to whether there are any un-encrypted passwords by issuing the following command from TACL and checking the first field in each record for an encrypted password:

```
TACL 1> FUP COPY $SYSTEM.SYSTEM.USERID,,SHARE,ASCII,NO HEAD
```

The shorter records at the end of the listing obtained contain remote passwords.

The auditor should determine whether password encryption is in effect.

NETWORK SECURITY

Tandem systems generally are not installed as stand-alone systems; instead, they are often integrated with other processing platforms in client environment. In addition to IBM systems, Tandem products support connectivity from Tandem systems to Digital Equipment Corporation (DEC) equipment, Honeywell systems, X.25 public networks, PC LANs, etc.

A Tandem EXPAND network security configuration, which supports Tandem-to-Tandem connectivity, is defined using key network attributes. These values determine how Tandem systems process incoming requests received from remote systems including personal computers.

Identifying Users on the Expand Network

In the EXPAND environment, users can be defined to the local environment and also be granted access to objects residing on other network nodes as well. A user is considered "local" with respect to the node on which a VERIFYUSER procedure call is issued by a process or an application.

Network nodes in a Tandem environment are identified in the form:

```
\SYS1
```

where SYS1 is the name of the network node. This node is used to remotely identify disk files on the node. For example, the disk file $VOL1.DATA.NAMES on remote node \SYS1, would be referenced as:

```
\SYS1.$VOL1.DATA.NAMES
```

The Tandem EXPAND network was designed with the philosophy that all remote nodes are equally distrusted, while within the local node there are trusted and cooperative entities. As a result, remote processes cannot suspend or activate local processes, generally cannot stop local processes, and cannot place a local process into debug mode.

Establishing network access to a remote node requires that several definitions be in place, as follows:

1. The user must be defined identically on any nodes for which access is desired. Thus user (30,4) named AUDIT.SMITH on a local node \SYS1 would also have to be defined as (30,4) and AUDIT.SMITH on node \SYS2 if he or she wanted access to \SYS2.

2. A remotepassword must be defined for each direction of access desired. If the remotepassword of NETPASS is defined for access from node \SYS1 to remote node \SYS2, then NETPASS must be defined on both \SYS1 and \SYS2. NETPASS only applies when accessing \SYS2 from \SYS1.

Access in the opposite direction from \SYS2 to \SYS1 would require that another remotepassword be defined to each node.

The remotepasswords described here are different from userid passwords that are entered by the user at signon time. The password is checked internally by the system when defined and is never entered subsequently by network users. The remotepassword only changes when it is explicitly modified by a REMOTEPASSWORD command.

3. The security string of the object to be accessed at the remote node must be set up so that the object can be remotely accessed by other network users.

To illustrate, Access between \LA and \SF

System \SF		System \LA	
Logon Name:	ADMIN.LARRY	Logon Name:	ADMIN.LARRY
Userid:	2,5	Userid	2,5
Remotepassword:\LA LJPASS		Remotepassword:\LA LJPASS	
Remotepassword:\SF MYPASS		Remotepassword:\SF MYPASS	

Definition of network access requires careful planning and coordination. Since userids and usernames must be defined identically across nodes, proper coordination of naming conventions is critical. Otherwise, a user could be defined to group number 3 which identifies purchasing on node \LA while also being defined to node \SF where group 3 is defined to payroll.

If default access to data is provided for group 3 members at both nodes and the group names match, the user could conceivably be granted access that is not appropriate. GUARDIAN security strings must be appropriately defined to ensure proper access control over network users.

*** Include graphic from Safeguard Administrators Manual ***
*** P/N 26192 Page 2-22 ***

The userid naming scheme should be reviewed by the auditor to determine whether proper coordination and conventions have been established across different nodes on the network. Since users must have the same userids across the network, conventions must be set such that the functions associated with a user's group name on one node does not have undesired implications on another node.

To examine the name of the alternate key file, enter

```
TACL 2> FUP INFO $SYSTEM.SYSTEM.USERID, DETAIL
```

and note the name of the alternate key file shown in the field entitled ALTFILE. There should be only one ALTFILE. The protection level of the alternate key file can be listed with the FILEINFO command.

The presence of remote passwords may be determined by the following command which will list the first fifty records:

```
TACL 3> FUP COPY $SYSTEM.SYSTERM.USERID,,SHARE,ASCII,NO HEAD
```

The shorter records at the end of the listing indicate whether any remote passwords are defined to any of the users.

Common Expand Network Capabilities

Generally speaking, a remote password is required in order to access another system on the network. The Tandem environment does, however, allow some basic capabilities for all network users across nodes without explicitly defining the user to the other nodes. The STATUS, FILEINFO, and PPD commands can be executed against other nodes on the network.

When issuing a FILEINFO command across a remote a node on the network, such as

```
TACL 1> FILEINFO \RMT1.$SYS*.*.*
```

to obtain information about a file, the user should keep in mind that the Tandem environment does not display volume and device names greater than six characters across nodes. Thus, the execution of a FILEINFO command might not provide a complete list of all files meeting the specified criteria. A file on node \RMT1 named $SYS1234 would not be shown on the remote node.

Dial-up Access Concerns

Special considerations exist in a dial up environment since the ability to access a system via a modem effectively negates some of the physical security aspects of the control environment established by the organization.

Dial-up access to Tandem systems can be controlled by defining the user to GUARDIAN authorization lists to authenticate users or by defining terminals and restricting their use to specific groups of users. The use of additional external system-wide passwords can be implemented as well as the implementation of dial back facilities, which call the user back at an authorized phone number that the system associates with that user, after authenticating the user.

Dial-up capabilities should be adequately restricted. Under GUARDIAN, a list of authorized dial-up users can be maintained and interrogated in the system's $CMON. If this method is used, care must be taken to ensure that $CMON is invoked during every system logon. Since TACL allows system logons without communicating with $CMON, controls built into $CMON may be easily circumvented by bypassing it and calling VERIFYUSER at the application level.

The CONFTEXT file contains parameters that describe the system hardware configuration. It is used as input to the SYSGEN process. In order to determine whether dial ups are running on the system with TACL, a listing of the CONFTEXT should be obtained. (See the topic SYSGEN PARAMETERS under System Customization.) The devices listed should be compared to that from a PUP LISTDEV. Devices with common dial up baud rates of 1200, 2400, 9600, or the parameter MODEM should be examined to confirm their appropriateness.

The auditor should review the installation's method of verifying dial up users, whether through an application, $CMON, GUARDIAN, or SAFEGUARD.

Deleting Network IDs

When establishing procedures to delete or update userids in the event of termination, resignation, or change of responsibilities, it is important to keep in mind that the Tandem user can be defined to multiple nodes in an EXPAND network.

When users leave an organization, exit procedures should be properly defined to ensure that managers on other systems throughout the network are appropriately informed so that the id is deleted from the system.

The auditor should examine the organization's personnel procedures when an employee is terminated or transferred to ensure that the network nodes, as well as the local system, are appropriately included in the communication chain.

Protection of Network Devices

Installations can define paths, as well as devices such as processors, I/O channels, and controllers, to the EXPAND network. (Protection of hardware devices is available only in a SAFEGUARD environment.)

In general, devices defined to the network should physically exist. That is, if the devices are not actually in use, then consideration should be given to removing them from the network configuration.

The PUP LISTDEV command in the format:

```
        TACL 1> PUP LISTDEV device-type, device-subtype
```

provides a list of configured network components as well as information describing whether the devices defined are active or inactive. A list of device types and subtypes is provided in the appendix.

Remote System Maintenance

Tandem Cyclone, CLX, and VLX environments have a Remote Maintenance Interface (RMI) feature which allows remote maintenance of the system by Tandem personnel. The RMI capability provides access to TACL and various diagnostic and monitoring functions as well as the ability to cold-load the system remotely from a system control panel. The Non Stop II, Nonstop TXP, and Nonstop EXT systems have similar functionality called the Operations and Service Processor (OSP).

Individuals who maintain the system remotely must have an RMI (or OSP) password to gain access. The RMI password is internal to RMI, is not controlled by GUARDIAN (or SAFEGUARD), and is initially defined when the Tandem hardware is installed. It is one to eight alphanumeric characters in length. Only super group members may change this password. The RMI and OSP passwords should be set to values that are consistent with the organization's password management policy.

The auditor should review the organization's password management policy to determine whether the RMI or OSP passwords are addressed.

Spooling Facilities

The spooler facility generally runs in the Tandem environment as process $S. The command interface for the spooler in the Tandem environment is called SPOOLCOM.

SPOOLCOM security may be checked with FILEINFO:

```
        TACL 1> FILEINFO $SYSTEM.*.SPOOLCOM
```

In order to determine the components that make up the spooler, list the name of the process in which the spooler supervisor code $SYSTEM.SYSTEM.SPOOL is running:

```
        TACL 1> STATUS *,PROG $SYSTEM.SYSTEM.SPOOL
```

```
Process    Pri    PFR  %WT  Userid  Program File          Hometerm
$SPLS  B   6,25   152  001  30,1    $SYSTEM.SYSTEM.SPOOL  $OPR1
$SPLS      7,25   152  001  30,1    $SYSTEM.SYSTEM.SPOOL  $OPR1
```

The example indicates that the program $SYSTEM.SYSTEM.SPOOL runs as process $SPLS. In order to examine the processes created by $SPLS, a temporary file is first created,

```
TACL 2> CREATE SPOOLCHK,50
TACL 3> PPD /OUT SPOOLCHK/
TACL 4> EDIT SPOOLCHK P SPCHKE
```

Resulting in,

```
TEXT EDITOR - T9602B30 - (08MAR87)
CURRENT FILE IS $FSD.LAJTEMP.SPCHKE
*
```

By entering the text editor,

```
*lb/$SPLS/
```

the following is obtained,

```
14    $SPLS      7,25       6,25       5,21
15    $S         6,24       7,26       $SPLS
16    $L         7,24       6,26       $SPLS
17    $SPRNT     4,24                  $SPLS
18    $SPNT3     1,40                  $SPLS
```

In this example, $S, $L, $SPRNT, and $SPNT3 are processes created by $SPLS. To check the security protecting these processes from TACL,

```
TACL 5> STATUS $S
```

which produces,

```
Process          Pri   PFR   %WT Userid   Program File   Hometerm
$S         6,24  149         005    30,1  $SYSTEM.SYSTEM.CSPOOL $OPR1
                             Swap File Name: $SYSTEM.#0092
$S    B    7,26  149         001    30,1  $SYSTEM.SYSTEM.CSPOOL $OPR1
                             Swap File Name: $SYSTEM.#0098
```

Here, the STATUS command shows that $SYSTEM.SYSTEM.CSPOOL is the actual program which runs under the process $S.

Finally, check the security surrounding this program,

```
TACL 6> FILEINFO $SYSTEM.SYSTEM.CSPOOL
```

and examine the RWEP string for the file. The process should be repeated for all processes created by $SPLS, such as $L, $SPRNT, and $SPNT3.

Other Tandem Network Security Considerations

As previously described, Tandem systems are generally networked both with other Tandem systems and systems from other vendors. Although this discussion has focused on network security considerations in Tandem-to-Tandem EXPAND networks, a variety of hardware and software products exists to facilitate connectivity with non-Tandem systems. These products have associated access control considerations.

Numerous products support system connectivity including the Tandem to IBM Link (TIL) and products from other vendors. Access control over these facilities may be implemented using a variety of physical, hardware, and software mechanisms. Installations must evaluate locally the requirement to implement encryption for network facilities. For many of these facilities, controls over the software process supporting the link, including the ability to initiate the process or communicate with it, are key to restricting the use of the facility.

Auditors must determine the specific facilities implemented in the installation and evaluate the appropriateness of the controls in place.

SECTION 5
SAFEGUARD SECURITY

SAFEGUARD is available to those installations which require extended security features but do not wish to build the features into the application. The implementation of security controls in a SAFEGUARD environment should, as a result, require less emphasis on controls developed by the individual installation.

The Safeguard access control security product is optional on Guardian systems. This is similar to the optional nature of RACF, CA-ACF2, and CA-TOP SECRET products in the IBM MVS operating system environment. SAFEGUARD runs as an extension of GUARDIAN security to provide additional, enhanced levels of protection.

Authentication:	SAFEGUARD allows for additional userid/password management rules.
Authorization:	Additional levels of protection are afforded over a larger number of entities including processes by use of Access Control Lists (ACL's).
Auditing:	Object access attempts, logon attempts, and security maintenance activity can be recorded in SAFEGUARD.
Administration:	SAFEGUARD allows for a decentralized administrative format, allowing for security groups. SAFECOM, the SAFEGUARD command interpreter, facilitates administrative duties. SAFEGUARD also allows the control and ownership of rights, such as who is allowed to administer SAFEGUARD, etc.

The following chart identifies the software programs within Safeguard and identifies the functional security mechanisms provided by each program.

Safeguard Access Control Security Features

FUNCTIONAL DESCRIPTION	PROGRAM	I&A	DAC	AUDIT	PRIV.	ADM.	TCB
Safeguard command interpreter. Provides user display, formatting services, and syntax parsing. Executes as a user application program.	SAFECOM	No	No	No	No	**YES**	**YES**
Safeguard Security Management Process. Security database maintenance, accepts parsed SAFECOM command messages or DSM/SPI tokenized messages. Controls user I&A for Safeguard defined terminals. Processes VERIFYUSER procedure calls for all other I&A. Audits subject I&A activity and Safeguard database maintenance. Synchronizes audit trail management for all SMON's. Executes as a fault-tolerant process pair.	OSMP	**YES**	No	**YES**	**YES**	**YES**	**YES**
Safeguard Security Monitor provides access mediation of objects using ACL's. Audits specified object accesses, and processes audit recording procedure calls from other Tandem subsystems. One OSMON ($ZS00-15) executes in each processor under the control of the OSMP F/T process pair.	OSMON	No	**YES**	**YES**	No	No	**YES**
Safeguard Audit Reduction Tool provides trusted audit record selection and basic record display functions. SAFEART executes as a user application program.	SAFEART	No	No	**YES**	No	No	**YES**
The Safeguard LOGON program is executed from the local system or a remote computer node to tell the OSMP to dynamically acquire the terminal from which LOGON was executed.	LOGON	**YES**	No	No	No	No	No
The SAFEACT program converts the new audit file format introduced in C22 Safeguard back to the previous C20 format. SAFEACT was provided as a migration to facilitate customer implementationof C22 Safeguard.	SAFEACT	No	No	No	No	No	No

Legend: User identification and authentication (I&A); ACL-based Discretionary Access Control (DAC) authorization; security relevant event auditing (AUDIT); use of privileges associated with security administrators, Super-Group and Group Managers (PRIV.); administration (ADM.); USA NCSC Trusted Computing Base evaluated component (TCB).

SYSTEM SECURITY PROTECTION

Basic Safeguard Resource Security

SAFEGUARD security protection records take precedence over the settings defined in GUARDIAN security strings. Whereas GUARDIAN protects system resources such as

$SYSTEM.SUBVOL1.FILE

at only the <u>diskfile</u> level with a "RWEP" security string, SAFEGUARD is capable of protecting the same resource using Access Control Lists (ACL's) at <u>multiple</u> levels including:

the volume level ($SYSTEM)
the subvolume level ($SYSTEM.SUBVOL1)
the file level ($SYSTEM.SUBVOL1.FILE)

Extended SAFEGUARD capabilities also apply to devices, such as $TAPE1 and subdevices, $TAPE1.#UNIT01. They can apply to processes, $S, and sub processes, $S.#LASER as well.

Care should be exercised with the assignment of access under GUARDIAN, since there may be situations when SAFEGUARD security might not be in effect and GUARDIAN security presides.

<u>The Effect of SAFEGUARD on GUARDIAN Security</u>

If the RWEP security string appears as "****", then the file is protected by SAFEGUARD on a file level. Interrogation of the level of protection then, must be done through the SAFECOM INFO DISKFILE command. However, a security string that does not appear as "****" under GUARDIAN does not necessarily imply that the file has not been afforded SAFEGUARD protection. It merely means that SAFEGUARD security has not been defined at the diskfile level, and that the file could be protected at the volume or subvolume level. The auditor would then have to interrogate the security settings at the volume or subvolume level to determine the level of protection given to the diskfile.

Security settings for diskfiles are physically maintained in the diskfile's file label. The label also contains a SAFEGUARD bit which is set when a SAFEGUARD record also exists at the <u>diskfile level</u>. While SAFEGUARD will always be referenced if it is installed in a Tandem environment, the bit does not get set if a SAFEGUARD record is created at the volume or subvolume level. Setting the bit to 1 results in the GUARDIAN security string to appear as "****" when interrogated.

<u>Recommended GUARDIAN Settings</u>

Sensitive files should be protected with "OOOO" settings. Permissive security settings, such as "AAAA" or "NNNN" should be discouraged. On a network system, the setting should be "UUUU".

Additional Protection Attributes

In addition to the standard GUARDIAN "RWEP" security string, SAFEGUARD adds CREATE and OWNER attributes. CREATE affords individuals the explicit ability to create specific named files or processes. OWNER allows the designation of ownership attributes to users who are not defined as the primary owner of a file.

The auditor should review the use of these attributes surrounding the organization's critical files to determine whether they are being utilized properly.

Default Protection of Objects

Requiring Access Control Lists

SAFEGUARD has an ACL-REQUIRED-DISKFILE attribute which if set would require that an access control record be present for a file before any type of access is granted.

In an installation where SAFEGUARD has been recently implemented, ACL-REQUIRED should only be added after the SAFEGUARD security environment has stabilized. If the option were set before adding any SAFEGUARD protection records, it would be possible to create the situation where no programs could run on the system, since no access control records are present.

Default Protection Strings

The SAFEGUARD DEFAULT-PROTECTION attribute, assignable at the individual userid level in SAFEGUARD, attaches a default access control list to every file created by that user. This ensures that any file created by that individual is defined to SAFEGUARD. Only the owner has the ability to add or change the DEFAULT-PROTECTION attribute.

ALL users defined to SAFEGUARD should have DEFAULT-PROTECTION to ensure that all files created by the user are defined to SAFEGUARD with a default access control list specified.

Under SAFEGUARD, a user's DEFAULT-PROTECTION setting may be examined with the SAFECOM command:

```
1> SAFECOM INFO USER userid, DETAIL
```

The default access control list for the users should be examined to ensure that the security is not too permissive.

Explicit Access Denials

The ACL also provides the capability of explicitly denying access to a resource with the DENY attribute. By using DENY, a user, including the super user, may be restricted from accessing a diskfile.

The DENY feature should be used with care, since DENY makes it possible to lock users out of a resource.

For example, the command

```
TACL 1>SAFECOM INFO DISKFILE $VOL1.SVOL2.DATA
```

would yield

```
                         LAST-MODIFIED          OWNER      STATUS

$VOL1.SVOL2 DATA         18OCT92, 10:20         22,4       THAWED

022,002                  DENY R,W,E,P,C,O
022,004                  R,W,E,P,O
```

In the example, the user 22,4 has been designated as the owner of the file $VOL1.SVOL2.DATA with read, write, execute, purge, and owner attributes. User 22,2, on the other hand, is explicitly denied access to the file.

Freezing System Resources

<u>Dangers of Freezing</u>

Freezing, or suspending userids in SAFEGUARD is an effective means of restricting access to the system for a specific userid when further investigation into that userid is required. On the other hand, it should be used with care, since it is possible to unintentionally freeze all users. Once a userid has been frozen, explicit action is required on the part of a security administrator, or super ID to thaw the userid. Installations which decide to utilize this feature should properly transition into doing so, after exploring alternative methods of controlling repeated access attempts, such as requiring a name logon.

<u>Typical Applications</u>

Typically, freezing of a userid might occur after an excessive number of unsuccessful logon attempts or when an extensive period of inactivity has passed. Similarly, a terminal device may be frozen so that users are restricted from using it to log on.

Userids can be frozen in a SAFEGUARD environment in several ways. The SAFECOM FREEZE USER command may be explicitly executed by the owner of a user to temporarily suspend user access, or a userid may be frozen as a result of consecutive failed logon attempts. Devices, such as terminals may also be frozen with the SAFECOM FREEZE DEVICE command.

The organization's security policy and guidelines should consider having userids frozen after a maximum number of unsuccessful logon attempts. Employees who take extended leaves of absence should have their userids frozen. High level userids, such as the super ID which are not being used for day-to-day activities should also be frozen.

A user's status may be determined by executing the SAFECOM INFO USER command. The SAFECOM INFO DEVICE command may be used to determine whether a device has been frozen.

SAFEGUARD Protection Records

Like GUARDIAN, SAFEGUARD authenticates users by interrogating the $SYSTEM.SYSTEM.USERID file. SAFEGUARD information and configuration parameters are maintained in the files:

```
$SYSTEM.SAFE.GUARD
$SYSTEM.SAFE.OTHER
$SYSTEM.SAFE.CONFIG
$volumename.SAFE.GUARD
```

Thus, each volume contains a SAFE.GUARD file containing the protection records for objects contained on that volume. $SYSTEM.SAFE.GUARD and $SYSTEM.SAFE.OTHER contain the various SAFEGUARD options in effect. Diskfile protection records are defined on each volume for both local and network users.

The files containing SAFEGUARD protection records and configuration should be properly protected so that only the super ID has access to the files listed above.

The SAFECOM command

```
=INFO DISKFILE $*.SAFE.GUARD
```

should be run to display the SAFEGUARD protection for all SAFE.GUARD files.

```
=INFO DISKFILE $SYSTEM.SAFE.OTHER
```

should be run to display the protection strings for the SAFE.OTHER file.

NOTE: The auditor should keep in mind when executing commands to view protection strings that security can exist at the volume, subvolume, or diskfile levels. Therefore, when no protection string is found when executing an INFO DISKFILE command, it does not necessarily mean that the data is not protected. Security can exist at the volume or subvolume level.

Clearing Deleted Files

When a file is deleted, the contents of the file are by default not erased and therefore are available to any programs which access the disk directly. The CLEARONPURGE option, described in the previous chapter, may be assigned and checked within SAFEGUARD by issuing the following SAFECOM command:

```
TACL 1> SAFECOM INFO DISKFILE filename, DETAIL
```

Temporary Users

Installations often have the need to establish temporary userids for students, contract programmers or software installers. Installations with educational facilities might reserve a block of userids for classroom use.

The USER-EXPIRES attribute in SAFEGUARD can be given to temporary users (e.g., contractors, or students in a class). This field contains the date that the userid will be frozen if no explicit action is taken by a security administrator.

The USER-EXPIRES attribute should always be used for temporary users, set to a date that coincides with the completion of the temporary user's need to access the system.

A user's expiration date may be determined by executing the SAFECOM INFO USER, DETAIL command.

SPECIAL PRIVILEGES

The Super ID

The super ID (255,255) is the most powerful userid available in the TANDEM environment. Proper management of the super ID is critical to the integrity of the Tandem environment, since the super ID has the ability to bypass security controls that are established for general users.

The command:

```
1>SAFECOM INFO USER 255,255, DETAIL
```

can be used in a SAFECOM environment to obtain information about the super ID.

Authorization to Define SAFEGUARD Security

In a SAFEGUARD environment, extended capability to define security records for objects is provided through OBJECTTYPE authority, which allows for the protection of a wider range of objects by a wider range of individuals.

Installations can choose to increase the number of individuals who can add users to the system using special access control lists called OBJECTTYPEs. Defining the ability to secure various elements within the Tandem SAFEGUARD environment may also be accomplished with OBJECTTYPEs.

OBJECTTYPE USER	Defines who can add users.
OBJECTTYPE DEVICE	Defines who can secure devices.
OBJECTTYPE SUBDEVICE	Defines who can secure subdevices.
OBJECTTYPE VOLUME	Defines who can secure disk volumes.
OBJECTTYPE SUBVOLUME	Defines who can secure sub-volumes.
OBJECTTYPE DISKFILE	Defines who can secure diskfiles.
OBJECTTYPE PROCESS	Defines who can secure processes
OBJECTTYPE SUBPROCESS	Defines who can secure sub-processes
OBJECTTYPE OBJECTTYPE	Defines who can create OBJECTTYPE authorization records.

By default, users have the ability to add diskfile records for files that they own.

OBJECTTYPE records should be defined for every type of object protected on the system. Without an OBJECTTYPE PROCESS for instance, users might have the capability to add SAFEGUARD records for a processes that they did not own.

OBJECTTYPE settings can be displayed using the INFO OBJECTTYPE command from SAFECOM.

```
TACL 1>SAFECOM INFO OBJECTTYPE nnnn, DETAIL
```

may be used to display OBJECTTYPE characteristics where nnnn is USER, DEVICE, SUBDEVICE, VOLUME, PROCESS, SUBPROCESS, OBJECTTYPE.

For example, the command,

```
TACL 1>SAFECOM INFO OBJECTTYPE USER, DETAIL
```

results in:

```
                   LAST-MODIFIED        OWNER          STATUS
USER
                   18JUN90, 14:27       250,1          THAWED

250,*              C

AUDIT-ACCESS-PASS = NONE               AUDIT-MANAGE-PASS = NONE
AUDIT-ACCESS-FAIL = NONE               AUDIT-MANAGE-FAIL = NONE
```

Here, all members of group 250 have the authority to add users.

Special Safeguard Groups

Under C22 and later versions of SAFEGUARD, an installation can associate a group with certain predefined attributes that help to facilitate administrative functions. These groups are the SECURITY-ADMINISTRATOR AND SYSTEM-OPERATOR groups.

The SECURITY-ADMINISTRATOR and SYSTEM-OPERATOR can be considered as a SAFEGUARD predefined set of attributes and capabilities which are explicitly associated with a group in SAFEGUARD when the group is added to the system. When adding a group using the ADD GROUP command, or modifying an existing group using the ALTER GROUP command, the individual adding the group can explicitly provide all group attributes, such as OWNER, AUDIT-ACCESS-PASS, AUDIT-ACCESS-FAIL, etc, or can adopt the attributes of another existing group.

Members of a group defined with the abilities of SECURITY ADMINISTRATOR can execute the following restricted commands:

```
ALTER SAFEGUARD
STOP SAFEGUARD
ADD AUDIT POOL
ALTER AUDIT POOL
ALTER AUDIT SERVICE
DELETE AUDIT POOL
SELECT
ADD TERMINAL
ALTER TERMINAL
DELETE TERMINAL
FREEZE TERMINAL
THAW TERMINAL
```

Under SYSTEM-OPERATOR, the group can execute the following restricted commands:

```
ADD AUDIT POOL
ALTER AUDIT POOL
DELETE AUDIT POOL
NEXTFILE
RELEASE
SELECT
FREEZE TERMINAL
THAW TERMINAL
```

Once these groups are defined, it is important to note that some of these capabilities, such as the ability to define security to a terminal, are actually taken away from the super ID and super groups. The size and members of these groups should be adequately restricted.

The auditor should inquire as to whether any groups have been defined as SECURITY-ADMINISTRATOR or SYSTEM-OPERATOR. The SAFECOM INFO GROUP command can be used to obtain additional information as to the group members.

The SAFECOM command,

```
=INFO GROUP SECURITY-ADMINISTRATOR
```

might yield the following information

```
                              LAST-MODIFIED       OWNER     STATUS

SECURITY-ADMINISTRATOR        18JUN92, 13:22      086,255   THAWED

086,002 DENY E,0
033,*        E,0
086,*        E,0
255,*        E,0
```

The owner of the group authorization record is user 86,255. All users in groups 33 and 255 are granted EXECUTE and OWNER privileges for the SECURITY-ADMINISTRATOR group. All users in group 86, with the exception of user 86,002, have EXECUTE and OWNER authority as well.

RESTRICTED FUNCTIONS

System Files

The SAFECOM INFO DISKFILE may be executed against the critical system files identified in the previous chapter to examine SAFEGUARD protection. In addition, several key SAFEGUARD files should be reviewed as well:

$SYSTEM.SYSnn.OSP	Object file for SMP
$SYSTEM.SYSnn.OSMON	Object file for SMON
$SYSTEM.SYSnn.SAFECOM	Object file for SAFECOM
$SYSTEM.SYSnn.SAFEART	Audit Reduction Tool
$SYSTEM.SYSnn.SAFEACT Format	Audit Conversion Tool C22-->C20
$SYSTEM.SYSnn.LOGON	

SYSTEM CUSTOMIZATION

Bringing Up the Security Environment

SAFEGUARD may be brought up in three different ways. First, it may be brought up interactively with a START SAFEGUARD command. Generally, this method is used during the early and initial stages of installation of the product. SAFEGUARD can also be brought up when the system is brought up by properly configuring the CIIN file. Finally, SAFEGUARD can be SYSGENed so that it starts when a Tandem system goes through its Initial Machine Load (IML).

SAFEGUARD should ultimately be SYSGENed so that it is always running when the system is running. The auditor should review the SYSGEN parameters and make a determination as to whether SAFEGUARD is SYSGENed to start up automatically at IML time. The SYSTEM_PROCESS_CODE_FILES section of the CONFTEXT file should include the OSMP and OSMON files to indicate that SAFEGUARD is included in system generation.

Controlling the Logon Process

SAFEGUARD and the Logon Process

Installations that have installed SAFEGUARD may use it to control system logons, bypassing TACL. AUTHENTICATE-MAXIMUM-ATTEMPTS allows the specification of maximum failed logon attempts before action is taken. That action can consist of freezing the userid (AUTHENTICATE-FAIL-FREEZE ON) or timing out for a time period specified by the AUTHENTICATE-FAIL-TIMEOUT attribute.

Giving SAFEGUARD control of terminals allows the use of other SAFEGUARD security features, such as a password grace period and a configurable command interpreter, plus the user can change the password at logon.

Single Logon Path

All system users should be required to sign onto the system using the same access path. That is, if SAFEGUARD is established by an installation to control the user authentication process, there should be no way for any users, whether general or high level super user, to use other methods to access the system, such as a terminal not defined to SAFEGUARD.

SAFEGUARD should be used whenever possible to control the logon process to take advantage of the various user identification controls that it has to offer. The auditor should determine whether there are alternative methods of logging onto the system. Also, determine whether there are any terminals defined to the system that are not defined to SAFEGUARD. The terminals defined to the system can be obtained from a CMI INFO LU * command, and the output compared to that of a SAFECOM INFO TERMINAL command.

A SAFECOM INFO DISKFILE $*.*.TACLCSTM may be run to determine the level of SAFEGUARD protection for the TACLCSTM files. The SAFECOM INFO USER 0,0 , DETAIL command or TACL USERS 0,0 command should be executed to determine whether the NULL.NULL user is defined to the system.

Auditing Attempts to Access System Resources

In the Tandem environment, the term "audit" in the context of a data file refers to the logging of attempted accesses to that file. GUARDIAN itself provides very limited audit features. Generally, if events are audited, the audit capabilities must be built by the user into either $CMON or the applications. The SAFEGUARD environment provides more extensive monitoring capabilities on several levels.

Monitoring of activity can be specified for an individual user's logon attempts, or it can be specified for an object, such as a diskfile. Specification of the conditions of the monitoring takes place by setting values in a number of parameters using SAFECOM. Two different categories of attempts to access system resources may be monitored within SAFEGUARD:

Nature of Action

ACCESS	The attempt to access the object
MANAGE	The attempt to alter the security settings in SAFEGUARD which control access to the object

Two types of results may be audited:

Success of Action

PASS	Successful attempts may be logged
FAIL	Unsuccessful attempts may be logged

Finally, audit capabilities may be selective across local and remote environments. That is, local and/or remote attempts may be audited.

Location of Action

NONE	No audit takes place
LOCAL	Audit local accesses only
REMOTE	Audit remote accesses only
ALL	Audit local and remote accesses

The nature, success, and location of the action are defined in concert to determine the level of activity monitoring which takes place. Additional details on how these actions are defined are described in the next few sections.

Auditing Logon Attempts

The AUDIT-ACCESS-PASS (successful attempts) and AUDIT-ACCESS-FAIL (unsuccessful attempts) attributes are assigned at the individual userid level and determine whether logon attempts are logged. Valid settings for these two attributes are NONE, LOCAL, REMOTE, and ALL.

Thus the SAFECOM command,

```
=INFO USER 5,255, DETAIL
```

would yield:

```
GROUP.USER          USER-ID OWNER    LAST-MODIFIED    LAST-LOGON STATUS

AUDIT.MGR  5,255  255,255 11OCT90, 8:10 31JAN92,11:21 THAWED


   USER-EXPIRES                          =        * NONE *
   PASSWORD-EXPIRES                      =        * NONE *
   PASSWORD-MAY-CHANGE                   =        * NONE *
   PASSWORD-MUST-CHANGE EVERY            =        30 DAYS
   FROZEN/THAWED                         =        THAWED
   STATIC FAILED LOGON COUNT             =        0

   AUDIT-ACCESS-PASS = NONE          AUDIT-MANAGE-PASS = NONE
   AUDIT-ACCESS-FAIL = LOCAL         AUDIT-MANAGE-FAIL = LOCAL
```

In this example, user 5,255 has the AUDIT-ACCESS-PASS attribute set to NONE, indicating that successful logon attempts are not logged. On the other hand, AUDIT-ACCESS-FAIL is set to LOCAL, indicating that all failed attempts to access the system locally will be logged.

Attempts to access the system using privileged userids should be monitored closely. The auditor should make a determination as to the installation's critical userids and ensure that attempts to use those ids are logged accordingly.

Auditing Access to User Authentication Records

SAFEGUARD allows for the logging of access to individual user authentication records. User authentication records contain information about a specific user, such as whether certain types of activity are logged for that individual, password management parameters, whether the id is frozen or thawed, etc. In short, the user authentication record contains the information which is shown with a SAFECOM INFO USER command.

The AUDIT-MANAGE-PASS (successful attempts) and AUDIT-MANAGE-FAIL (unsuccessful attempts) attributes define the logging of attempts to modify user authentication records. Valid settings are ALL, LOCAL, REMOTE, or NONE.

Again, using the same example described earlier, the SAFECOM command,

```
=INFO USER 5,255, DETAIL
```

would yield:

```
GROUP.USER        USER-ID OWNER    LAST-MODIFIED    LAST-LOGON STATUS

AUDIT.MGR  5,255  255,255 11OCT90, 8:10 31JAN92,11:21 THAWED


        USER-EXPIRES                      =       * NONE *
        PASSWORD-EXPIRES                  =       * NONE *
        PASSWORD-MAY-CHANGE               =       * NONE *
        PASSWORD-MUST-CHANGE EVERY        =       30 DAYS
        FROZEN/THAWED                     =       THAWED
        STATIC FAILED LOGON COUNT         =       0

        AUDIT-ACCESS-PASS = NONE          AUDIT-MANAGE-PASS = NONE
        AUDIT-ACCESS-FAIL = LOCAL         AUDIT-MANAGE-FAIL = LOCAL
```

For the userid 5,255, only local failed attempts are logged. Typically, attempts to access information surrounding critical userids should be monitored closely. Attempts that are logged when this attribute is set include attempts to user a SAFECOM INFO USER command, attempts to suspend or activate a userid with a FREEZE USER or THAW USER command, or simply attempts to modify any attributes associated with a userid.

Auditing Attempts to Access Objects

Auditing can be defined under SAFEGUARD at the object level, where a valid object includes a diskfile, subvolume, volume, process, device, etc. The level of auditing in effect is determined by attributes which are set within an object authorization record.

The AUDIT-ACCESS-PASS and AUDIT-ACCESS-FAIL attributes determine whether successful and/or unsuccessful attempts to access the object are logged. Valid settings are ALL, LOCAL, REMOTE, or NONE.

The command,

```
        TACL 1>SAFECOM INFO DISKFILE $VOL1.SVOL2.DATA, DETAIL
```

would yield

```
                                 LAST-MODIFIED          OWNER          STATUS

$VOL1.SVOL2 DATA                 18OCT92, 10:20         22,4           THAWED

022,002                          DENY
022,004                          R,W,E,P,O

AUDIT-ACCESS-PASS = LOCAL                     AUDIT-MANAGE-PASS = NONE
AUDIT-ACCESS-FAIL = NONE                      AUDIT-MANAGE-FAIL = NONE

LICENSE = OFF       PROGID=OFF                CLEARONPURGE = OFF
```

The example illustrates that AUDIT-ACCESS-PASS is set to LOCAL and AUDIT-ACCESS-FAIL is NONE for the diskfile $VOL1.VOL2.DATA, meaning that all successful local attempts to access the file are logged, but not the unsuccessful attempts.

Attempts to access an installation's critical objects should be closely monitored. The auditor should obtain a list of those critical objects, generally identified in a risk assessment, and identify the level of auditing that is in effect through the SAFECOM INFO command.

Auditing Access to Object Authentication Records

Like the user authentication records described earlier, object authentication records contain critical information concerning the protection of that particular object. Information such as the users who have the authority to access the object is maintained, as well as the level of auditing which is in effect.

AUDIT-MANAGE-PASS and AUDIT-MANAGE-FAIL are the attributes which determine whether successful or unsuccessful attempts to access these records are logged. Valid settings are ALL, LOCAL, REMOTE, and NONE.

If we again refer to the previous example,

The command,

```
        TACL 1>SAFECOM INFO DISKFILE $VOL1.SVOL2.DATA, DETAIL
```

would yield

```
                          LAST-MODIFIED            OWNER              STATUS
$VOL1.SVOL2 DATA          18OCT92, 10:20           22,4               THAWED

022,002                   DENY
022,004                   R,W,E,P,O

AUDIT-ACCESS-PASS = LOCAL              AUDIT-MANAGE-PASS = NONE
AUDIT-ACCESS-FAIL = NONE               AUDIT-MANAGE-FAIL = NONE

LICENSE = OFF     PROGID=OFF           CLEARONPURGE = OFF
```

AUDIT-MANAGE-PASS is set to NONE, as is AUDIT-MANAGE-FAIL. Thus, no logging is taking place for modifications to the protection record for the diskfile $VOL1.VOL2.DATA. Attempts to use an INFO DISKFILE command to obtain information about the diskfile protection, attempts to freeze or thaw the diskfile, or attempts to change any attributes stored within the object protection record will not be audited.

Attempts to access the protection records for an installation's critical files, processes, or devices should be closely monitored. The auditor should obtain a list of those objects, generally identified in a risk assessment, and identify the level of auditing that is in effect through the SAFECOM INFO command.

Recommended Level of Audit

Requirements for the level of logging which takes place for a given installation will vary, depending upon the organization's risk assessment. This section discusses some suggested settings that an installation should consider as a guideline.

An audit record should be created for all attempts to logon to the system whether successful or unsuccessful. Similarly, attempts to change object authorization records and user authentication records should be closely monitored by setting the appropriate settings to ALL or LOCAL for a non-networked system. In this case, AUDIT-ACCESS-PASS and AUDIT-ACCESS-FAIL would be set to ALL or LOCAL. This would also allow for the audit of failed logon attempts to undefined userids.

Although generally there is no need to audit access attempts to all objects on a system, it is important to establish monitoring of all critical objects as determined by the organization's risk assessment.

One possible set of recommended audit settings might be summarized as follows:

	Logging of Attempts:	
Event	Successful	Unsuccessful
All signon attempts	X	X
Access to General Objects		X
Protection Records of General Objects		X
Access to Critical Objects	X	X
Protection Records of Critical Objects	X	X

These settings would be implemented as follows:

User Signon

AUDIT-AUTHENTICATE-PASS	ALL	Log all successful logon attempts.
AUDIT-AUTHENTICATE-FAIL	ALL	Log all unsuccessful logon attempts.

Critical Objects

AUDIT-ACCESS-PASS	ALL	Log successful access attempts to objects.
AUDIT-ACCESS-FAIL	ALL	Log unsuccessful access attempts to objects.
AUDIT-MANAGE-PASS	ALL	Log successful access attempts to protection records of object.
AUDIT-MANAGE-FAIL	ALL	Log unsuccessful access attempts to protection records of object.

General Objects

AUDIT-ACCESS-PASS	NONE	Do not log successful access attempts to objects.
AUDIT-ACCESS-FAIL	ALL	Log unsuccessful access attempts to objects.
AUDIT-MANAGE-PASS	NONE	Do not log successful access attempts to protection records of object.
AUDIT-MANAGE-FAIL	ALL	Log unsuccessful access attempts to protection records of object.

NOTE: On a system which is not located on an EXPAND network, LOCAL is equivalent to ALL.

A SAFECOM INFO DISKFILE may be run against the installation's critical files to determine the logging of activity which is taking place against a file. Global settings may be obtained via the SAFECOM INFO SAFEGUARD, DETAIL command.

Audit Record Management

Audit pools are the subvolumes that contain files that contain the audit records of activity which has taken place during the logon process, or during access to system resources.

SAFEGUARD Audit Pools

SAFEGUARD audit records under the C22 and above releases are written to audit files located in pools. If the installation does not define a subvolume containing files for the audit records to be written to, the system will store the records in the $SYSTEM.SAFE subvolume.

Installation Defined Audit Pools

In addition, an installation may define its own audit pools. When defining its own audit pools, the installation may also define the action to be taken when a file overflow condition is encountered, as well as action to be taken when the audit pool is inaccessible for any reason.

All audit pool files on the main operating system volume, i.e.,"$SYSTEM.SAFE", files should be protected, as well as any installation defined audit pools; such that, only appropriate staff may access the files. When audit files overflow, the installation should have appropriate procedures to review the contents of the file, as well as utilize a different file to record subsequent activity.

The audit file containing the audit records should be monitored closely and appropriate action taken when it becomes full.

The SAFECOM INFO DISKFILE command should be run using the "*" wild card character to locate all SAFEGUARD protection records for the audit pools on all disk volumes,

```
$*.SAFE.*
```

The security strings shown should be examined to ensure that the RWEP string is appropriate. A FILEINFO $*.SAFE.* should also be performed from TACL to ensure that the audit records have a security string of "----", indicating that all users are prohibited from access with the exception of the super ID.

If the installation defines its own audit pools, the SAFECOM INFO AUDIT POOL $volumename.subvolume may be specified to determine the number of files and extents associated with the audit pool. INFO AUDIT SERVICE may be used to determine the current status of the audit pool, the next audit pool to be used, and the recovery action, which is the action that takes place on an overflow or inaccessible condition.

```
=INFO AUDIT SERVICE
```

results in

```
CURRENT AUDIT POOL            $secure.audit2

CURRENT AUDIT FILE            A0000003

NEXT AUDIT POOL               $keeper.audit1

RECOVERY                      DENY GRANTS

CURRENT STATE                 NORMAL

WRITE-THROUGH CACHE           ON

EOF REFRESH                   ON
```

In the previous example, $secure.audit2.A0000003 is the current file being used for audit records. $keeper.audit1 will be used when $secure.audit2 is completed. The recovery action of DENY GRANTS denies the granting of authorization and authentication requests for which auditing is required which could in effect, freeze the system.

Password Management Features

The SAFECOM command,

```
=INFO SAFEGUARD
```

command is used to identify SAFEGUARD's password management options.

<u>User Expiration</u>

SAFEGUARD provides extended password management abilities over GUARDIAN. Under SAFEGUARD, an expiration date may be associated with a given userid (USER-EXPIRES) so that action is taken, such as freezing the user, when the date specified in USER-EXPIRES is reached.

USER-EXPIRES should always be specified for temporary users, such as contract programmers, or students in a class so that the ids may not be used once their work is complete.

Password Change Interval

Passwords can be required to be changed at periodic intervals (PASSWORD-MUST-CHANGE). A PASSWORD-MUST-CHANGE = 90 would prompt the user to change his/her password every 90 days. If no action is taken, the userid is frozen. SAFEGUARD warns the user and forces the user to change the password. The length of this grace period can be limited.

PASSWORD-MUST-CHANGE should be set at appropriate intervals to suit the organization's security policy. An organization might also require that high level userids, such as the super user and super group members, change passwords more frequently than less powerful user IDs. Many Tandem customers require users to change their passwords with intervals ranging from 30 days to 90 days. Very few customers require password change intervals of less than 30 days for any system user.

Password Change Window

A time window may be specified to explicitly allow users to change passwords (PASSWORD-MAY-CHANGE) prior to the expiration period defined in PASSWORD-MUST-CHANGE. Thus, a PASSWORD-MAY-CHANGE of 10 would allow the alteration of a password 10 days prior to expiration. Finally, a grace period can be specified, allowing a user a predetermined amount of time which a password may be changed AFTER the expiration date has passed.

PASSWORD-MAY-CHANGE can be set to only allow changing of passwords one week prior to the expiration date. This would prevent users from selecting new passwords, and immediately changing the passwords back to their original values.

Password Length and Failed Attempts

The minimum number of characters required in a password can be specified (PASSWORD-MINIMUM-LENGTH). Consecutive failed logon attempts (AUTHENTICATE-MAXIMUM-ATTEMPTS) can trigger specific action to guard against attempts to guess passwords. This action can consist of suspending a process or freezing a userid (AUTHENTICATE-FAIL-FREEZE = ON) when the maximum attempts are exceeded. A logon session may also timeout (AUTHENTICATE-FAIL-TIMEOUT). In addition, a "blind password" option is available, to prevent passwords from being displayed on the screen as they are typed.

PASSWORD-MINIMUM-LENGTH should be set to an appropriate value of 6 to reduce the possibilities of guessing passwords or establishing single character passwords.

AUTHENTICATE-MAXIMUM-ATTEMPTS should be set to a value of 3 and either AUTHENTICATE-FAIL-FREEZE or AUTHENTICATE-FAIL-TIMEOUT should be considered to discourage repeated attempts at unauthorized access. NOTE: A disadvantage to specifying AUTHENTICATE FAIL FREEZE when maximum attempts at access are exceeded is that an unscrupulous user may freeze any userid, including the super ID or supergroup members, by exceeding maximum attempts. The only way to recover from this would be to tape cold-load the system. This possibility should be considered carefully before an installation chooses to exercise these SAFEGUARD options.

Logging on As Another User

The ability for the super ID or a group manager to logon as another user without knowing that user's id can be controlled by setting PASSWORD-REQUIRED to ON. PASSWORD-REQUIRED should be set to ON so that the super ID and group managers cannot sign on as another user without knowing that individual's userid.

Password History

SAFEGUARD maintains a history of passwords (PASSWORD-HISTORY) so that the same password cannot be used over and over again. If the password entered by the user has been used in the past and is retained by the system password history, the user is asked to enter an alternative password.

PASSWORD-HISTORY should be set at a high enough value, such as 32, to discourage the frequent repetition of passwords.

Recommended Settings

Execute the following

```
1> SAFECOM INFO SAFEGUARD
```

to obtain the values of numerous password options within SAFEGUARD. Recommended password management options might be summarized as follows:

Password Management Feature	SAFEGUARD	Desired Setting
Userid Expiration	USER-EXPIRES	60 days
Mandatory Password Change	PASSWORD-MUST-CHANGE	30
Minimum Password Length	PASSWORD-MINIMUM-LENGTH	6
Maximum Unsuccessful Logon Attempts	AUTHENTICATE-MAXIMUM-ATTEMPTS	3
Action Taken When Maximum Logon Attempts is Exceeded	AUTHENTICATE-FAIL-FREEZE AUTHENTICATE-FAIL-TIMEOUT	NO 180
Super ID and Group. Manager signon as another user	PASSWORD-REQUIRED	ON
Password History	PASSWORD-HISTORY	32

Encryption of Passwords

PASSWORD-ENCRYPT instructs SAFEGUARD to encrypt passwords prior to saving them. It encrypts passwords from the point at which PASSWORD-ENCRYPT is set to ON. Any passwords residing on the system PRIOR to setting PASSWORD-ENCRYPT are not encrypted. The encryption is one way, so that passwords may be verified, but not decrypted.

In SAFEGUARD environments, the SAFECOM INFO SAFEGUARD command can be executed and the PASSWORD-ENCRYPT parameter may be examined to ensure that encryption of passwords is taking place.

SAFEGUARD Conversion Issues

Once extended security capabilities are adopted by implementing SAFEGUARD, there are a number of GUARDIAN functions that are no longer needed to maintain the security environment and should be adequately restricted, since they could potentially be used as alternative avenues to gain unauthorized access to the system.

To ensure that users can only be maintained using SAFEGUARD, the GUARDIAN ADDUSER, DELUSER, and RPASSWRD programs should be protected with

OWNER=SUPER.SUPER and FROZEN. These programs are used in maintaining GUARDIAN security, and are no longer needed, since SAFEGUARD provides the extended management abilities. Use of the RPASSWRD program, for example, would successfully define a remote password, but SAFEGUARD would have no record of the change.

If SAFEGUARD has been implemented at a Tandem installation, the auditor should determine whether these functions are available and how they are protected. The FILEINFO command may be used to confirm their protection string.

Authorization Search Sequence

Under SAFEGUARD, the installation has the ability to define resource protection on numerous levels, in contrast to GUARDIAN, where the only level of protection is provided on a diskfile level. Since the definition of a diskfile consists of several components, the volume, the subvolume, and the filename, and security under SAFEGUARD can be defined for each component, it is possible to define conflicting security configurations for a single resource.

As a result, the order, or authorization search sequence, in which the security for these components is checked determines whether access should be allowed for a resource.

In other words, it is possible to define security so that access is granted to a specific diskfile, such as $VOLUME.SUBVOL.DISKFILE while denying access to the volume $VOLUME. As a result, the issue of whether an access request is granted or denied depends heavily on the order in which access is checked, as well as the number of levels for which access is checked.

Suppose that an access request for the file $VOLUME.SUBVOL. DISKFILE is being verified for a user, (42,7) through SAFEGUARD.

1. **Direction of Search:**

   The DIRECTION parameters determine which ACL is interrogated FIRST when determining access to an object. DIRECTION parameters are defined for several entities:

   DIRECTION-DISKFILE
   DIRECTION-PROCESS
   DIRECTION-DEVICE

   If DIRECTION-DISKFILE is set to FILENAME-FIRST, security for the individual file is verified first. If security is not specified for the specific filename ($VOLUME.SUBVOL. DISKFILE), the subvolume security ($VOLUME.SUBVOL) would be checked, followed by the volume ($VOLUME). If DIRECTION-DISKFILE is set to VOLUME-FIRST, then the search sequence is reversed, checking volume first, followed by subvolume, followed by diskfile.

Similarly, protection of processes may be defined at the process, or subprocess level. Setting DIRECTION-PROCESS to PROCESS-FIRST would interrogate process level security at the process first, followed by the subprocess. The opposite search sequence would take place if DIRECTION-PROCESS were set to SUBPROCESS-FIRST.

Devices can be protected at the device or subdevice level. Thus, if DIRECTION-DEVICE were set to DEVICE-FIRST, security for the device would be checked, followed by the subdevice, and vice versa for SUBDEVICE-FIRST.

2. **Length of Search**

The COMBINATION parameter determines when access checking is stopped.

COMBINATION-DISKFILE
COMBINATION-PROCESS
COMBINATION-DEVICE

If COMBINATION-DISKFILE is set to FIRST-ACL, then the search stops when the first access control list is encountered. If access is explicitly defined for the user (access is defined for a specific user, such as 42,7), then access is granted. Otherwise, access is denied.

If FIRST-RULE is specified, the search continues until a definitive ruling is provided in reference to the user attempting access. In this case, explicitly granting access (to 42,7; 42,*; or *.*) or explicitly DENYing access.

Finally, ALL specifies that all ACLs for the object are evaluated. In order for access to be granted, access must be provided at all levels. For example, if access were granted to user 42,7 at the volume and subvolume level, but was denied at the filename level, access would be denied.

3. **Scope of Search**

CHECK defines whether a particular level of security is checked or bypassed.

CHECK-VOLUME
CHECK-SUBVOLUME
CHECK-FILENAME
CHECK-PROCESS
CHECK-SUBPROCESS
CHECK-DEVICE
CHECK-SUBDEVICE

IF CHECK-FILENAME is OFF, then the filename security is ignored. Thus, if user 42,7 were granted access at the volume and subvolume level, but denied access at the filename level, but CHECK-FILENAME were set to off, the denial at the filename level would be ignored.

If no protection record is specified at any level for the object, then the GUARDIAN security string is used.

Careful consideration must be taken when defining the search sequence parameters, along with the organization's naming conventions for volumes and subvolumes. The sequence defined should be examined to ensure that the access given is consistent with the organization's security policy.

The SAFECOM INFO SAFEGUARD command can be executed to determine the settings which comprise the authorization search sequence for the installation.

Disk File Authorization Search Sequence Table

This table illustrates the interaction of Safeguard options DIRECTION-DISKFILE, CHECK-VOLUME, CHECK-SUBVOLUME, CHECK-FILENAME, and COMBINATION-DISKFILE in granting or denying access to disk files. These options are set using the ALTER SAFEGUARD command and can be displayed using the INFO SAFEGUARD command.

In the table, LEVEL refers to the direction in which Safeguard searches the for the protection records containing the Access Control Lists (ACLs). The search direction is determined by the Safeguard configuration attribute DIRECTION-DISKFILE, which may be set to either VOLUME-FIRST or DISKFILE-FIRST.

VOLUME-FIRST	**Object Type**	**DISKFILE-FIRST**
Level 1	VOLUME	Level 3
Level 2	SUBVOLUME	Level 2
Level 3	DISK FILE	Level 1

For example, if the direction is VOLUME-FIRST, the volume protection record is read and its' ACL is searched first, the subvolume ACL second, and the diskfile ACL third. If the search direction is DISKFILE-FIRST, the diskfile protection record ACL is searched first, the subvolume second, and the volume third.

The access evaluation is affected by the settings of CHECK-VOLUME, CHECK-SUBVOLUME, and CHECK-FILENAME; which may cause one or more of the levels to be ignored as follows:

CHECK-VOLUME {ON or OFF} which controls whether the volume protection record for the specific disk volume will be read and its' ACL used for access decisions.

CHECK-SUBVOLUME {ON or OFF} which controls whether the subvolume protection record for the specific disk volume will be read and its' ACL used for access decisions.

CHECK-FILENAME {ON or OFF} which controls whether the subvolume protection record for the specific disk volume will be read and its' ACL used for access decisions.

The evaluation of disk file access rules also depends on the COMBINATION-DISKFILE Safeguard option, whenever two or more of the checking levels are "ON", i.e., CHECK-VOLUME, CHECK-SUBVOLUME, and CHECK-FILENAME. COMBINATION-DISKFILE has the following optional authorization search sequences:

FIRST-RULE means that Safeguard uses the first protection record ACL that contains the specified user id to make the access decision.

FIRST-ACL means that Safeguard uses the first protection record ACL encountered to make the access decision.

ALL means that Safeguard uses all available protection record ACLs to make the access decision.

The following abbreviations are used in the table:

Abbreviation	Meaning	Abbreviation	Meaning
Y	ACL evaluates to Yes	Permit	Access Permitted
N	ACL evaluates to No	Deny	Access Denied
NM	ACL contains No Mention	G90	Guardian 90 rules apply
NR	No protection record exists at this level		

Disk File Access Evaluation Table

Levels:			Combinations/Evaluations:		
First	Second	Third	FIRST-ACL	FIRST-RULE	ALL
Y	Y	Y	Permit	Permit	Permit
Y	Y	N	Permit	Permit	Deny
Y	Y	NM	Permit	Permit	Deny
Y	Y	NR	Permit	Permit	Permit
Y	N	Y	Permit	Permit	Deny
Y	N	N	Permit	Permit	Deny
Y	N	NM	Permit	Permit	Deny
Y	N	NR	Permit	Permit	Deny
Y	NM	Y	Permit	Permit	Deny
Y	NM	N	Permit	Permit	Deny
Y	NM	NM	Permit	Permit	Deny
Y	NM	NR	Permit	Permit	Deny
Y	NR	Y	Permit	Permit	Permit
Y	NR	N	Permit	Permit	Deny
Y	NR	NM	Permit	Permit	Deny
Y	NR	NR	Permit	Permit	Permit
N	Y	Y	Deny	Deny	Deny
N	Y	N	Deny	Deny	Deny
N	Y	NM	Deny	Deny	Deny
N	Y	NR	Deny	Deny	Deny
N	N	Y	Deny	Deny	Deny
N	N	N	Deny	Deny	Deny
N	N	NM	Deny	Deny	Deny
N	N	NR	Deny	Deny	Deny

Levels:			Combinations/Evaluations:		
First	Second	Third	FIRST-ACL	FIRST-RULE	ALL
N	NM	Y	Deny	Deny	Deny
N	NM	N	Deny	Deny	Deny
N	NM	NM	Deny	Deny	Deny
N	NM	NR	Deny	Deny	Deny
N	NR	Y	Deny	Deny	Deny
N	NR	N	Deny	Deny	Deny
N	NR	NM	Deny	Deny	Deny
N	NR	NR	Deny	Deny	Deny
NM	Y	Y	Deny	Permit	Deny
NM	Y	N	Deny	Permit	Deny
NM	Y	NM	Deny	Permit	Deny
NM	Y	NR	Deny	Permit	Deny
NM	NR	Y	Deny	Permit	Deny
NM	NR	N	Deny	Deny	Deny
NM	NR	NM	Deny	Deny	Deny
NM	NR	NR	Deny	Deny	Deny
NR	Y	Y	Permit	Permit	Permit
NR	Y	N	Permit	Permit	Deny
NR	Y	NM	Permit	Permit	Deny
NR	Y	NR	Permit	Permit	Permit
NR	N	Y	Deny	Deny	Deny
NR	N	N	Deny	Deny	Deny
NR	N	NM	Deny	Deny	Deny
NR	N	NR	Deny	Deny	Deny
NR	NM	Y	Deny	Permit	Deny
NR	NM	N	Deny	Deny	Deny
NR	NM	NM	Deny	Deny	Deny
NR	NM	NR	Deny	Deny	Deny
NR	NR	Y	Permit	Permit	Permit
NR	NR	N	Deny	Deny	Deny
NR	NR	NM	Deny	Deny	Deny
NR	NR	NR	G90*	G90*	G90*

Note:

1. * Indicates that access is denied if ACL-REQUIRED option is YES.
2. This table was reprinted from the Safeguard Reference Manual (P/N 26191) Appendix E.

NETWORK SECURITY

Identifying Users on the EXPAND Network

The SAFECOM INFO USER,DETAIL command may be executed to identify whether a user has a remotepassword and what node he has access to. This command will only work for the owner of the user's authentication record or the super ID.

SECTION 6
AUDIT PROGRAM / CHECKLISTS

The audit program is designed to allow the auditor to review either a TANDEM environment using security features provided by the GUARDIAN operating system or one with the SAFEGUARD access control package. The audit program is separated into GUARDIAN and SAFEGUARD sections. Auditors reviewing SAFEGUARD controls however, should also consult the GUARDIAN sections, as many GUARDIAN features are still in effect in SAFEGUARD environments.

Audit Userid Requirements

A userid is needed to gather system information during the audit. Since an auditor userid with "read all access" is not available, the holder of the super ID or super group ID will need to help gather audit information.

Documenting an Understanding of the Organization

- Obtain organization charts for the company. Using the organization charts, document which individuals are directly associated with security administration. Organization charts can also be used to help determine whether user profiles, group profiles and ACL's appear appropriate, in line with the organization structure and employee's job function.

- Obtain Information Systems department security policies and procedures. Review the policies to determine their coverage. Use them to develop specific audit steps to address the procedures.

- To gain an understanding of the system configuration, document the hardware and software environment. This should include hardware models, operating system version, network configuration, application software, primary programming languages, databases, etc. Some TACL commands that can be issued to gather this information include:

  1> PUP LISTDEV (to list devices)

  2> FILEINFO $*.*.* (to list ALL files)

  3> FILEINFO $SYSTEM.SYS*.* (to list of files in the $SYSTEM.SYSTEM and $SYSTEM.SYSnn subvolumes, see note 1.)

  4> VPROC $SYSTEM.SYSTEM.* & VPROC $SYSTEM.SYSnn.*

(to list version procedures time stamps for these subvolumes, see note 1.)

Note 1: There may be several different Guardian 90 operating system images available on the computer system. These images are numbered from SYS00 (i.e. "zero-zero") to SYS99. These versions many exist to implement computer operations contingency plans, new software or hardware migration, etc. Computer operations should be able to identify the reason for each operating system image, and the current executing system image. The executing system image can usually be identified by using the TACL STATUS * command, which will display the processes currently executing on the system. The resulting display will contain a number of processes executing from file $SYSTEM.SYSnn.OSIMAGE, where SYSnn is the current operating system subvolume.

SYSTEM SECURITY PROTECTION

Basic Diskfile Security

- Review policy and procedures to determine whether the organization has a documented procedure for assigning and modifying access rights to resources.

- Identify resource ownership assignment and determine whether it follows the organization's policy and structure.

- Identify the organization's files that contain critical data.

- Review the protection surrounding the organization's critical data.

- Ascertain whether there is a periodic review and recertification by resource owners and security administration for access assignments.

Default Protection of Objects

- Ascertain whether there is a policy for setting default protection over diskfiles to prevent it from being set too loosely, potentially allowing unnecessary access.

- Check default security settings by listing key system users and checking settings other than "OOOO" (only owner has RWEP capabilities).

Tape Security

- To maintain security protection over backed up and restored files ascertain whether the organization requires the use of "NOMYID" attribute when the BACKUP process is issued.

Process Security

- Identify the organization's critical processes and determine whether backup processes should be running.

Clearing Deleted Files

- Determine whether the organization's critical files have CLEARONPURGE set.

Detection of Orphaned Files

- Ascertain whether there are procedures to ensure that files are deleted, or ownership is modified, when an employee who owns these files is transferred or terminated.

- When userids are reused, ensure that there is a procedure to ensure that ownership of files is not mistakenly inherited by the new user.

- Check the userids of former employees to determine whether ownership was properly removed or transferred.

SPECIAL PRIVILEGES

User Identification

- Ascertain whether the organization has a policy for the assignment of userids and access authorization that is consistent with the user's job responsibilities.

- Determine whether there are policies regarding userid naming conventions and their group assignment.

- Ascertain whether there is a policy that each system user must be assigned a unique ID for accountability purposes, and that userids cannot be shared.

- The organization should prevent users from using their userid number to logon to the system. This can be verified by reviewing the logon TACL to see whether the NAMELOGON TACL flag is set.

User Classification

- Ascertain whether the super ID, super group, group manager users are assigned in accordance with job responsibilities and the customer's security policy.

- Ascertain whether all remaining users are assigned in accordance with job responsibilities and the customer's security policy.

- Ascertain whether the NULL.NULL user ID has been reviewed and is appropriately controlled in accordance with the customer's security policy.

User Authentication

- If applications are present that perform their own VERIFYUSER call, check the code to ensure that VERIFYUSER is used correctly.

  Notes: The VERIFYUSER procedure call can potentially be used to create automated password guessing schemes. However either GUARDIAN 90 authentication attempt failure delays; or SAFEGUARD authentication attempt delays, user ID freezing, and authentication failure auditing provide preventive and detective controls.

Authorization to Define Security to Objects

- Review access to the ADDUSER and DELUSER programs to determine whether access to them is appropriate.

  Notes: these two programs contain internal logic that allows only group managers (i.e. user IDs with nnn,255) to add or delete users within their own group. These programs are not used on SAFEGUARD equipped systems, and should be "frozen" or otherwise controlled by SAFEGUARD to prevent accidental use.

RESTRICTED FUNCTIONS

System Files

- Verify who has access to system files and critical utilities; review for appropriateness.

- Verify that GUARDIAN 90 operating system software and other Tandem products in $SYSTEM.SYS* are traceable to a Tandem provided Software Update Tape (SUT).

- Verify that third-party software vendor entries in system subvolumes are appropriate and traceable to the corresponding vendor.

- Compare the contents of the $SYSTEM.SYSTEM and $SYSTEM.SYSnn subvolumes for duplicate entries.

Licensing

- Ascertain whether the organization has policies regarding program file licensing, including approvals, documentation, segregation of duties over licensing, source code review and testing, monitoring, etc.

- Review the installation's licensed programs to determine their appropriateness. User programs should not be licensed.

- Determine who can execute the licensed programs.

EXTENSIONS AND MODIFICATIONS

PROGID Programs

- Ascertain whether the organization has a formal review process for PROGID programs, and whether there is a policy against PROGIDing system files.
- List out all PROGID programs on each volume and who can execute.
- Review the PROGID programs and the appropriateness of who can execute them.

Inspect and Debug

- Review the organization's security policy to see whether the use of INSPECT and DEBUG is restricted to proper use.

  Note:INSPECT and DEBUG restrict users to programs that they own. Only the super-super user Id is allowed to INSPECT or DEBUG programs owned by other users.

$CMON Generalized User Exit

- If the $CMON exit is documented, review the documentation to determine the appropriateness of the logic. Also review the program code for $CMON for appropriateness.
- Verify if $CMON is executing presently.
- Review security over $CMON to ensure that only authorized users have access.

SYSTEM CUSTOMIZATION

SYSGEN Parameters

- Ascertain whether the organization has change control procedures over modifications to the CONFTEXT and CONFAUX parameters.
- Determine whether access to the CONFTEXT and CONFAUX files is appropriate.
- Determine whether the super ID is set up to ignore explicit security denials.

Controlling the Logon Process

- Determine the security protection over TACLCSTM and TACLLOCL.

- Check if NULL.NULL is defined by issuing the following command:

```
3> USERS 0,0
```

- Review the following TACL bind options to determine whether they have been activated.

```
AUTOLOGOFFDELAY
LOGOFFSCREENCLEAR
REMOTECMONREQUIRED
REMOTECMONTIMEOUT
REMOTEsuper ID
```

- Review the following ALTER SAFEGUARD options are set to determine if the $CMON process is called during SAFEGUARD user authentication.

```
CMONREQUIRED
CMONTIMEOUT
```

Password Management Features

- Ascertain whether passwords are required by policy. Test the requirements for passwords by reviewing the minimum length of the password.

- Ascertain whether the following options are set using the bind options.

  PROMPTPASSWORD
  BLINDPASSWORD
  MINPASSWORDLEN
  ENCRYPTPASSWORD

- Identify whether the user file which contains passwords is properly protected.

Encryption of Passwords

- Verify that all passwords are encrypted.

Auditing Access the System and System Resources

- When SAFEGUARD audit logging is used, ascertain whether audit files are appropriately protected and periodically reported. Assess the audit report review procedures, i.e., frequency, level, follow-up. Ascertain how audit files are managed to prevent loss of audit event information.

- Ascertain whether $CMON is used to audit TACL events, such as RUN commands and LOGON attempts.

  Note: $CMON is only invoked for a limited number of TACL functions. Guardian 90 does not invoke $CMON, therefore it cannot be used to record access to system resources, such as diskfiles, processes, etc. Security information, such as clear text passwords, are not passed to $CMON.

Auditing System "Console" Activity

- Ascertain whether system console or application system events are logged.

Database Audit Trail

- Determine whether critical database files can be appropriately recovered or reconstructed from available information sources.

- Determine whether the GUARDIAN 90 Transaction Monitoring Facility (TMF) is used to provide database forward and backward recovery. Review the operational policies, procedures and controls for TMF and access to the TMF disk and tape recovery files. These files contain database record "Before" and "After" images.

NETWORK SECURITY

Identifying Users on the Expand Network

- Ascertain whether there are formal procedures for setting and maintaining remote passwords.

- Determine whether proper coordination and naming conventions have been established across different nodes on the network to ensure that functions associated with a user's group name on one node do not have different implications on another node.

Dial-up Access Concerns

- To determine if dial-ups are defined, issue the following TACL commands:

  ```
  1> FILEINFO $SYSTEM.*.CONF*

  2> FUP COPY $SYSTEM.SYS00.CONFTEXT
  ```

  Compare the devices listed to those on a PUP LISTDEV listing. Devices with common dial-up baud rates of 1200 or 2400 should be examined to confirm their appropriateness.

- Review the installation's method of verifying dial-up users, whether through an application, $CMON, GUARDIAN, SAFEGUARD, etc.

- Review policies for the use of dial-up access. Are there additional access controls other than TANDEM-related, such as dial-back or additional levels of password control?

Deleting Network IDS

- Examine the organization's procedures that address employee separations or transfers to ensure accesses on the network nodes and the local system are appropriate to their job responsibilities.

Remote System Maintenance

- Review the organization's password management standards to determine whether the RMI or OSP passwords are included.

Remote Duplicate Database Facility (RDF)

- Ascertain who has access to RDF files.

Spooling Facilities

- Determine if multiple spooler subsystems are being run for enhanced parallel performance or information confidentiality. Each spooler subsystem should be reviewed according to the installation's security policy with respect to the sensitivity of the reports handled by that spooler. These reviews should include verifying appropriate protection of the individual spooler datafiles.

- Determine who has access to SPOOLCOM and identify the protection of the components, which make up each spooler.

SAFEGUARD AUDIT PROGRAM

SYSTEM SECURITY PROTECTION

Basic Safeguard Resource Security

- Ascertain whether the organization has a documented procedure for assigning and modifying access rights to resources, such as, volumes, subvolumes, disk files, processes, devices, etc.

- Identify resource ownership assignment and determine whether it follows the organization's policy and structure.

- Identify the organization's files containing critical or financially significant data and their protection.

- Assess whether there is a periodic review and recertification by resource owners and security administration for access assignments.

- Determine if the ACL-REQUIRED-DISKFILE (or PROCESS, DEVICE, VOLUME, etc.) parameter is on.

- Ascertain whether there is a policy for use of the "DENY" feature, which locks users from resources.

Additional Protection Attributes

- Determine whether CREATE and OWNER attributes are in effect at the computer system and whether they are assigned according to one's job responsibilities.

Default Protection of Objects

- Ascertain whether there is a policy for setting default protection over resources to prevent it from being set too loosely, and potentially allowing unnecessary access.

- Check default protection of system resources by listing system users and examining the user's DEFAULT-PROTECTION to make sure it is not set too loosely.

Freezing of Resources

- Determine whether freezing is in effect and whether the installation is at risk from locking out all users.

SAFEGUARD Protection Records

- Verify that only the super ID has access to SAFEGUARD protection records or configuration parameters, which are maintained in the following files:

  $SYSTEM.SAFE.GUARD
  $SYSTEM.SAFE.OTHER
  $SYSTEM.SAFE.CONFIG
  $SYSTEM.SAFE.CONFIGA
  $volumename.SAFE.GUARD

Clearing Deleted Files

- When the organization has highly sensitive files, consideration should be given to assigning the CLEARONPURGE option to those files. Determine if any disk files have this option.

SPECIAL PRIVILEGES

User Identification

Authorization to Define Security to Objects

- Ascertain whether there is a policy regarding the assignment of SAFEGUARD OBJECTTYPE authorization to appropriate individuals. Assess whether the OBJECTTYPE authorities have been properly issued.

Special Safeguard Groups

- Determine whether there is a policy regarding the implementation of the special SAFEGUARD SECURITY-ADMINISTRATOR AND SECURITY-OPERATOR groups.

- Assess whether SECURITY-ADMINISTRATOR AND SECURITY-OPERATOR group membership has been appropriately defined based on site security policies, based on the job requirements of the designated group members..

SYSTEM CUSTOMIZATION

Bringing Up the Security Environment

- Ascertain whether SAFEGUARD is started automatically as part of the GUARDIAN 90 Cold-Load procedure or manually activated and deactivated.

- Ascertain whether the super ID is set up to ignore explicit security denials.

Auditing Attempts to Access System Resources

- Evaluate whether attempts to access system resources are adequate and appropriate.

- Check critical files to determine whether the following logging features are activated by setting the listed SAFECOM attributes:

  AUDIT-ACCESS-PASS
  AUDIT-ACCESS-FAIL
  AUDIT-MANAGE-PASS
  AUDIT-MANAGE-FAIL

Auditing Logon Attempts

- Evaluate whether attempts to logon to the system are appropriately logged for privileged system users.

Audit Record Management

- Determine whether the installation's audit pools are properly managed and protected from unauthorized access.

Password Management Features

- Review the following SAFEGUARD subsystem attributes for password parameters with respect to the security policy. The setting status can be listed by issuing the SAFECOM INFO SAFEGUARD command.

  USER-EXPIRES
  PASSWORD-MUST-CHANGE
  PASSWORD-MAY-CHANGE
  PASSWORD-EXPIRY-GRACE
  PASSWORD-MINIMUM-LENGTH
  PASSWORD-HISTORY
  PASSWORD-ENCRYPT

- Some password management attributes can also be specified on an individual user basis. Review individual user information for conformance to the security policy concerning:

  USER-EXPIRATION
  PASSWORD-MUST-CHANGE
  PASSWORD-EXPIRY-GRACE

  This information can be listed for all users by the SAFECOM INFO USER *.* command.

- Identify whether the user file which contains passwords is properly protected.

SAFEGUARD Conversion Issues

- Ensure that the GUARDIAN ADDUSER, DELUSER, and RPASSWRD programs are properly deleted or protected from use by SAFEGUARD access control lists.

- The PASSWORD change program might optionally be protected from use if all terminals are controlled by the SAFEGUARD Authentication Service, which allows users to change their passwords as part of the LOGON process.

Authorization Search Sequence

- Review the DIRECTION, COMBINATION, and CHECK parameters to see whether they are consistent with the implementation of ACL security.

APPENDIX A

ACCESS NEEDED TO REVIEW THE TANDEM ENVIRONMENT

The audit procedures in this guide require a high level of access capability to objects in the Tandem environment. Because security in a Tandem environment is defined at the resource level, as opposed to the individual user ID level, it might not be feasible to create entries in each access control list for an auditor. Definition of a standard user ID for the auditor without explicitly providing access to all system files and critical application files will not provide the auditor adequate information to provide a control assessment.

As a result, much of the audit work might have to performed under the supervision of the super ID or super group member, using existing high level user IDs.

KEY COMMANDS

The auditor will utilize a number of key commands in gathering information about the Tandem environment.

GUARDIAN ENVIRONMENTS

IDENTIFYING USERS

A GUARDIAN user may be identified with the TACL USERS command. The command may be issued in the form

```
TACL 1>   USERS groupid,userid

                 or

TACL 1>   USERS groupname.username
```

For example,

```
TACL 1>      USERS SUPER.*
```

results in:

```
    GROUP         USER I.D. #        SECURITY    DEFAULT VOLUMEID

    SUPER       .MARY 255,015         OOOO        $SPOOL.MARY
    SUPER       .CARL 255,200         AAAA        $SPOOL.CARL
    SUPER      .SPOOL 255,030         AAAA      $SPOOL.SPOOLER
```

IDENTIFYING DISKFILE INFORMATION

Information may be obtained about a diskfile through the FILEINFO command in the form,

```
TACL 1> FILEINFO diskfile-name
```

For example,

```
TACL 1> FILEINFO NOTES
$BOOKS1.LSWORK
```

results in:

```
     CODE           EOF      LAST MODIFICATION      OWNER     RWEP
   NOTES 101       21484     10-APR-90  15:16:56    147,36   "AOAO"
```

SAFEGUARD ENVIRONMENTS

IDENTIFYING USERS

A SAFEGUARD user may be identified with the SAFECOM INFO USERS command. The command may be issued in the form

```
TACL 1>SAFECOM INFO USERS groupid,userid, DETAIL
```

or

```
TACL 1>SAFECOM INFO USERS groupname.username, DETAIL
```

For example,

```
=INFO USER 5,255, DETAIL
```

would yield:

```
GROUP.USER   USER-ID OWNER    LAST-MODIFIED      LAST-LOGON     STATUS

 AUDIT.MGR     5,255  255,255    11OCT90, 8:10      31JAN92,11:21   THAWE
                                                                      D


USER-EXPIRES                              =      * NONE *
PASSWORD-EXPIRES                          =      * NONE *
PASSWORD-MAY-CHANGE                       =      * NONE *
PASSWORD-MUST-CHANGE EVERY                =      30 DAYS
FROZEN/THAWED                             =      THAWED
STATIC FAILED LOGON COUNT                 =      0

AUDIT-ACCESS-PASS = NONE                  AUDIT-MANAGE-PASS = NONE
AUDIT-ACCESS-FAIL = LOCAL                 AUDIT-MANAGE-FAIL = LOCAL
```

IDENTIFYING DISKFILE INFORMATION

Information may be obtained about any resource through the SAFECOM INFO command in the form,

```
TACL 1>SAFECOM INFO resourcetype resourcename
```

For example, the command

```
1>SAFECOM INFO DISKFILE $VOL1.SVOL2.DATA
```

would yield:

```
                    LAST-MODIFIED  OWNER         STATUS
$VOL1.SVOL2
DATA                18OCT92, 10:20 22,4          THAWED

022,002             DENY
022,004             R,W,E,P,O
```

<end of document>