Part IV
Technical Controls

Overview of Part IV

This section focuses on security controls that are executed by the system. These controls are dependent upon the proper functioning of the system for their effectiveness. The implementation of technical controls, however, always requires significant operational considerations -- and must be consistent with the management of security within the organization. Chapters in this section include:

- Identification and Authentication
- Access Control
- Audit
- Cryptography

The Identification and Authentication (I&A) chapter discusses how I&A supports security by providing a means of ascertaining and verifying who a user (either human or process) is. I&A is a fundamental building block of computer security since it is the basis for most types of access control and for establishing *user accountability*. For example, *Auditing* of computer activity relies on the knowledge of who users of a computer system really are. Basic principles of I&A are presented, followed by a review of current technology used in I&A. The problems of passwords are examined in some detail.

Next, logical access controls are discussed. They provide a technical means to control the use of information and other system resources. This allows control over what information users can utilize, the programs they can run, and the modifications they can make. Criteria that can be used to determine what users (or processes) will be granted to specific resources are discussed. This is followed by an examination of the various technical implementation of logical access controls, including *access control lists*, *constrained user interfaces*, and *security labels*. Ways to control interactions between the system and outside systems and services and administration of access controls are also examined.

The following chapter, Audit Trails, focuses on how auditing can be used to be able to determine who has performed what actions on a system. This capability is useful in order to deter, catch, and recover from inappropriate actions. Auditing benefits a security program and can be used to help achieve individual accountability, to reconstruct events after system-based malfunctions, detect system and network intrusions, and analyze problems. The chapter reviews the types of audit logs (system, application and user), issues involved in determining what should be recorded in an audit trail, and how often (and by whom) the data should be reviewed. The strong ties between auditing and I&A and logical access controls are stressed.

The final Handbook chapter involves cryptography, the encoding and decoding of information. Cryptography is a useful mechanism in protecting the confidentiality and integrity of data through the use of *encryption* and *electronic signatures*, respectively. The chapter reviews the two basic classes of cryptography, *secret key* and *public key*, and the benefits and drawbacks of each. Uses of cryptography in support of user authentication are presented as are issues which must be confronted in administering cryptographic-based applications.

16 Identification & Authentication

16.1 Introduction

Computer systems use identification and authentication to ascertain and verify who a user (either human or a process) is.[62] Identification and authentication (I&A) is a critical building block of computer security since it is the basis for most types of access control and for establishing user accountability.[63] I&A is also, for most systems, the first line of defence. It is a technical measure that prevents unauthorized people (or unauthorized systems) from entering a computer system. This chapter will discuss the basic principles of identification and authentication, the current technology used to provide I&A, and will discuss some implementation issues.

The ability to identify users is critical to many types of *access control* and for establishing *user accountability*. Many types of access control require that the system be able to differentiate among users. For example, access control is often based on *least privilege*, which refers to the granting to users of only those accesses required to perform their duties. This requires that users be identified to the system.

User accountability is the property that enables activities on a system be traced to individuals who may then be held responsible for their actions. It requires the linking of activities on a computer system with a specific individual and, therefore, requires the system to identify users.

Identification and authentication is both the claim by a user to be a specific person known to the system and the verification that the user is who they claim to be. Identification is the means by which a user *provides* their identity to the system. Authentication is the means of establishing the *validity* of the user's claimed identity. Typically, these tasks are separated for implementation reasons. (See example in box.)

A typical user identification could be JSMITH (for Jane Smith). This information can be known by system administrators and other system users. A typical user authentication could be Jane Smith's password which is kept secret. This way system administrators can set up Jane's access and see her activity on the audit trail and system users can send her email, but no one can pretend to be Jane (unless they discover her password).

16.2 Basic Principles

Secure I&A is, unfortunately, difficult to achieve on most computer systems. Computer

62 Computers also use authentication to verify that a message or file has not been altered and to verify that a message originated with a certain person. This chapter only addresses user authentication. The other forms of authentication are addressed in the Cryptography Chapter.

63 Not all types of access control require identification and authentication.

systems recognize people based on authentication data *as received by* the computer system. This presents two challenges: collecting authentication data and transmitting authentication data securely. A third challenge is knowing whether the person who was originally authenticated is *still* the person using the computer system. For example, someone may walk away from the computer while still logged on and a second person starts using the account.

There are three means of authenticating a user's identity to the computer system, which can be used alone or in combination:

- Something the individual *knows,* (a secret) e.g., a password, Personal Identification Number (PIN), or cryptographic key;

- Something the individual *possesses,* (a token) e.g., an ATM card or a smart card;

- Something the individual *is,* (a biometric) e.g., behavioral characteristics such as a voice pattern or handwriting dynamics or physiological characteristics such as a fingerprint.

> For most applications, tradeoffs will need to be made between security, ease of use, and ease of administration especially in modern networked environments.

While it may appear that each of these means could provide strong authentication, there are problems associated with each method. If someone wants to pretend to be someone else on a computer system, they can guess or learn that individual's password. They can steal or fabricate tokens. Each method also has drawbacks for the legitimate users and system administrators. Users forget passwords and may lose tokens. Administrative overhead keeping track of I&A data and tokens can be substantial. Biometric systems have significant technical, user acceptance and cost problems as well.

16.3 Current Technology

This section explains the current technologies and the benefits and drawbacks of the I&A technologies discussed earlier. Although the section is divided into the three means of authentication (something the user knows, something the user possesses, and something the user is), the reader should keep in mind that *many of the technologies use more that one means*. The three means provide, however, an understandable framework for explaining authentication. Many of the technologies make use of cryptography because it can significantly strengthen authentication. The chapter, though, does not explain how cryptographic techniques work. The Cryptography Chapter of the Handbook provides an overview of cryptography, but detailed technical explanations are not provided. The references listed in this chapter and the Cryptography Chapter explain the technical details.

16.3.1 Authentication Based on Something the User Knows

The most common form of authentication is passwords. This technique is based solely on something the user knows. There are other techniques besides *conventional* passwords that are based on knowledge, such as knowledge of a cryptographic key.

> Although conventional passwords are the most common type of authentication, there are other techniques based on something the user knows.

16.3.1.1 Passwords

In general password systems work by requiring the user to enter a userid and a password (or passphrase or PIN). The system compares the password to a previously stored password for that userid. If there is a match, the user is authenticated.

Benefits of Passwords. Passwords have been successfully providing security for computer systems for a long time. They are integrated into many operating systems and user and system administrators are familiar with them. When properly managed in a controlled environment, they can provide effective security. However, as the long list below shows, password systems have significant problems based on technical limitations of the systems and the management of the systems.

Problems with Passwords. There are significant problems with passwords. The security of a password system is dependent upon the ability to keep passwords secret, and there are many ways that the secret is divulged. Password security can be increased with user training and careful system administration, however, there is no fix for the problem of electronic monitoring. In order to solve the problem, more advanced authentication must be used (based on cryptographic techniques or tokens). Some of the significant problems are discussed:

1. Guessing or finding passwords. Because users have to remember passwords, they tend to make them easy to remember. That often makes them easy to guess. The names of people's children and pets or favorite sports teams are common passwords which are easy to guess. If passwords are hard to remember or a user has many passwords, people often write them down where they can be found. Many computer systems are shipped with administrative accounts that have preset passwords. Because these passwords are standard, attackers can use them to break into systems. People can observe someone entering a password or PIN. The observation can be done by someone in the same room or by someone some distance away using binoculars. This is often referred to as *shoulder surfing*.

2. Giving passwords away. Users may share their passwords. They may give their password to a coworker in order to share files. In addition, people can be tricked into divulging their passwords. This process is referred to as *social engineering*.

3. Brute force. There are several brute force attacks on passwords that involve either the use of an on-line dictionary or an exhaustive attempt at different character combinations.

4. *Electronic monitoring*. When passwords are transmitted to a computer system, they can be electronically monitored. This can happen on the network used to transmit the password or on the computer system. Encrypting the password does not solve this problem because encrypting the same password will create the same ciphertext; the ciphertext becomes the password.

5. *Accessing the password file*. If the password file is not protected by strong access controls, the file can be downloaded. Even if the file is encrypted, brute force can be used to guess passwords.

Passwords Used as Access Control. Some mainframe operating systems and many PC applications use passwords as a means of restricting access to specific resources within a system. Instead of using mechanisms such as access control lists (see the Access Control Chapter), access is granted by entering a password. The result is a proliferation of passwords that can reduce the overall security of a system. While the use of passwords as a means of access control is common, it is an approach that is often less than optimal and not cost-effective.

Hints for Increasing Password Security

Many of these techniques are discussed in Federal Information Processing Standard (FIPS) 112, *Password Usage*.

Password generators. If users are not allowed to generate their own passwords, they cannot pick easy to guess passwords. Some generators create only pronounceable passwords to help users remember them. However, users tend to write down hard to remember passwords. See FIPS 181, *Automated Password Generator*.

Limits on login attempts. Many operating systems can be configured to lock a userid after a set number of failed login attempts. This helps to prevent brute force guessing of passwords.

Password attributes. Users can be instructed or the system can force them to select passwords with a certain minimum length, with special characters, which are unrelated to their userid, or to pick passwords which are not in an online dictionary. This makes passwords harder to guess.

Changing passwords. Periodic changing of passwords can reduce the damage done by stolen passwords and make brute force attempts to break into system more difficult. Too frequent changes, however, can be irritating to users.

Technical protection of the password file. Access control and one-way encryption can be used to protect the password file itself.

Educating users. One of the most important methods of increasing password security is to teach users how to protect their passwords.

16.3.1.2 Cryptographic Keys

Although the authentication derived from the knowledge of a cryptographic key may be based entirely on something the user knows, it is necessary for the user to also possess (or have access to) something which can perform the cryptographic computations. For this reason, the protocols used are discussed in the Smart Tokens Section below. However, it is possible to implement these types of protocols without using a smart token. Additional discussion is also provided under the Single Login Section.

16.3.2 Authentication Based on Something the User Possesses

Although some techniques are based solely on something the user possesses, most of the techniques described below are a combination with something the user knows. This combination can provide significantly stronger security than either something the user knows or possesses alone.

Objects that a user possesses for the purpose of authentication are called *tokens*. This section divides tokens into two categories: *memory tokens* and *smart tokens*.

16.3.2.1 Memory Tokens

Memory tokens store, but do not process, information. Special reader/writer devices control the writing and reading of data to and from the tokens. The most common type of memory token is a magnetic striped card, in which a thin stripe of magnetic material is affixed to the surface of a card (e.g., stripes on the back of credit cards). The most common application of memory tokens for authentication to a computer system is ATM cards. These use a combination of something the user possesses (the card) with something the user knows (the PIN).

There are computer systems authentications based solely on possession of a token, but they are less common. Token-only systems are more likely to be used in other applications, such as physical access. (See the Physical & Environmental Controls Chapter.)

Benefits of Memory Token Systems. Memory tokens when used with PINs provide significantly more security than passwords. For a hacker or other would-be masquerader to pretend to be someone else, the hacker must have both a valid token *and* the corresponding PIN. This is much more difficult than obtaining a valid password and userid combination (especially since most userid's are common knowledge). In addition, memory cards are inexpensive to produce.

Problems with Memory Token Systems. Although there are sophisticated technical attacks made against memory token systems, most of the problems associated with them relate to their cost, the administration of the system, user dissatisfaction, and the compromise of PINs. Most of the techniques for increasing the security of memory token systems relate to the protection of PINs. These are discussed in the sidebar on Helpful Hints for Improving Password Security.

1. Requires special reader. The first problem with a memory card is that it requires a reader. The user cannot simply type in a password. This increases the cost of their use. The readers used for memory tokens must include both the physical unit that reads the card and a processor that determines whether the card and/or the PIN entered with the card is valid. If the PIN or token is validated by a processor that is not physically located with the reader, then the authentication data is vulnerable to electronic monitoring.

2. *Token loss*. Tokens can be lost, stolen, or forged. A lost token may prevent the user from being able to login in until a replacement is provided. This can increase administrative overhead costs.

> Attacks on memory cards system have sometimes been quite creative. One group built a fake ATM machine which they installed at a local shopping mall. The machine collected valid account numbers and corresponding PINs which the thieves used to forge cards. The forged cards were then used to withdraw money from authentic ATMs.

The lost token can be found by someone who wants to break into the system or the token can be stolen or forged. If the token is also used with a PIN, any of the methods described above in password problems can be used to obtain the PIN. Common methods are finding the PIN taped to the card or observing the PIN being entered by the legitimate user. In addition, any information stored on the magnetic stripe that has not been encrypted can be read.

3. *User Dissatisfaction*. In general, users want the computer system to be easy to use. Many users find having to carry and present a token to be inconvenient. Two-step authentication is more time-consuming than simply entering a password. However, user dissatisfaction may be reduced if users see the need for increased security.

16.3.2.2 Smart Tokens

A smart token expands the functionality of a memory token by incorporating one or more integrated circuits that perform the functions of a computer. When used for authentication, a smart token is another example of authentication based on something a user possesses (i.e., the token itself). A smart token typically requires a user also provide something the user knows (i.e., a PIN or password) in order to "unlock" or use the smart token.

There are many different types of smart tokens. In general smart tokens can be divided three different ways based on: physical characteristics, interface, and protocols used. These three divisions are not mutually exclusive.

Physical Characteristics. Smart tokens can be divided into two groups: *smart cards* and other types of tokens. A smart card looks like a credit card, but incorporates an embedded microprocessor. Smart cards are defined by an International Standards Organization (ISO) standard. Smart tokens that are not smart cards can look like calculators or keys or other small portable objects.

Interface. Smart tokens have either a manual or an electronic interface. Manual or human interface tokens have displays and/or keypads to allow humans to communicate with the card. Smart tokens with electronic interfaces must be read by special reader/writers. ISO smart cards, described above, have an electronic interface. Smart tokens which look like calculators usually have a manual interface.

Protocol. There are many possible protocols that a smart token can use. In general, they can be divided into three categories: static password exchange, dynamic password generators, and challenge-response. The first type of tokens work similarly to memory tokens, except that the users authenticate themselves *to the token* and then the token authenticates the user to the computer.

> These protocols can also be used without a token if the user can use their local workstation to perform the computations.

A token that uses a *dynamic password generator* protocol creates a unique value, for example, an 8-digit number, which changes periodically (e.g., every minute). If the token has a manual interface, the user simply reads the current value and then types it into the computer system for authentication. If the token has an electronic interface, the transfer is done automatically. If the correct value is provided, the login is permitted and the user is granted access to the system.
Tokens that use a *challenge-response* protocol work by having the computer generate a challenge, such as a random string of numbers. The smart token then generates a response based on the challenge. This is sent back to the computer, which authenticates the user based on the response. The challenge-response protocol is based on cryptography. Challenge-response tokens can use either electronic or manual interfaces.

There are other types of protocols, some more sophisticated and some less sophisticated. The three types described above are the most common types.

Benefits of Smart Tokens

> Smart tokens offer great flexibility and can be used to solve many authentication problems.

The benefits of smart tokens vary depending on the type used. In general, they provide greater security than memory cards. Smart tokens can be used to solve the problem of electronic monitoring even if the authentication is done across an open network. The solution uses *one-time passwords*.

1. One-time passwords. Smart tokens which use either dynamic password generation or challenge-response protocols can create one-time passwords. Electronic monitoring is not a problem with one-time passwords because each time the user is authenticated to the computer, a different "password" is used. (A hacker can learn the one-time password through electronic monitoring, but it is of no value to the hacker.)

2. Reduced risk of forgery. Smart tokens are much harder to forge than memory tokens. The memory on a smart token is, generally, not readable unless the PIN is entered. In addition, the tokens are more complex and, therefore, more difficult to fabricate correctly.

3. Multi-application. Smart tokens with electronic interfaces, such as smart cards, provide a way for users to access many computers using many networks with only one login. This is further discussed in the Single Login Section below. In addition, a single smart card can be used for multiple functions such as physical access or as a debit card.

Problems with Smart Tokens

Like memory tokens, most of the problems associated with smart tokens relate to their cost, the administration of the system, and user dissatisfaction. Smart tokens are generally less vulnerable to the compromise of PINs because the authentication usually takes place on the card. (It is possible, of course, for someone to watch a PIN being entered and steal that card.) Smart tokens cost more than memory cards because they are more complex. Challenge-response calculators cost significantly more than memory cards.

*1. Need reader/writers **or** human intervention.* Smart tokens can use either an electronic or human interface. An electronic interface requires a reader, which creates additional expense. Human interfaces require more actions from the user. This is especially true for challenge-response tokens with a manual interface, which require the user to type the challenge into the smart token and the response into the computer. This can increase user dissatisfaction.

> Electronic reader/writers can take many forms such as a slot in a PC or a separate external device. Most human interfaces consist of a keypad and display.

2. Administration. Smart tokens, like passwords and memory tokens, require strong administration. For tokens which use cryptography, this includes key management. (The Cryptography Chapter addresses key management.)

16.3.3 Authentication Based on Something the User Is

Biometric authentication technologies employ unique characteristics (or attributes) of an individual to authenticate that person's identity. Biometric systems can employ physiological attributes (such as fingerprints, hand geometry, or retina patterns) or behavioral attributes (such as a voice patterns and hand-written signatures). Biometric authentication technologies based upon these attributes have been developed for computer login applications.

> Biometric authentication is technically complex, expensive, and user acceptance can be difficult. However, advances continue to be made to make the technology more reliable, less costly, and more user-friendly.

Biometric authentication generally operates in the following manner:

- Prior to any authentication attempts, a user is "enrolled" by creating a reference profile (or template) based on the desired physical attribute. The resulting template is associated with the identity of the user and stored for later use.

- When attempting to be authenticated, the user's biometric attribute is then measured.

- The previously stored reference profile of the biometric attribute is then compared with the measured profile of the attribute taken from the user. The result of the comparison is then used to either accept or reject the user.

Biometric systems can provide an increased level of security for computer systems, but the technology is still less mature than that of memory tokens or smart tokens. Imperfections in biometric authentication devices arise from technical difficulties in measuring and profiling physical attributes as well as from the somewhat variable nature of physical attributes. Many physical attributes change depending on various conditions. For example, a person's speech pattern may change under stressful conditions or when suffering from a sore throat or cold.

Due to their relatively high cost, biometric systems are typically used with other authentication means in environments requiring high security.

16.4 Implementing Authentication Systems

This section addresses some of the important implementation issues for authentication systems including: administration, maintaining authentication, and single login.

> *Administration* addresses the secure creation, distribution, and storage of authentication data.
>
> *Maintaining authentication* addresses the problem of someone pretending to be someone else *after* that person has been authenticated.
>
> *Single login* address some methods for reducing the burden of identification and authentication of users.

16.4.1 Administration

Authentication systems need to create, distribute and store authentication data. For passwords, this includes creating passwords, giving them to users, and maintaining a password file. Token systems involve the creation and distribution of tokens and PINs and data that tells the computer how to recognize valid tokens and PINs. For biometric systems, this includes creating and storing profiles.

> Administration of authentication data is a critical element for all types of authentication systems. The administrative overhead associated with I&A can be significant.

The administrative side of creating and distributing authentication data and tokens can be a large job. Identification data must be kept current by adding new users and deleting former users. If the distribution of passwords or tokens is not controlled, system administrators will not know if they have been given to someone other than the legitimate user. It is critical that the distribution system ensure that authentication data is firmly linked with a given individual. Some of these issues are discussed in the Personnel/User Chapter under User Administration.

In addition, identification and authentication administration should address lost or stolen password or tokens. It is often necessary to monitor systems to look for stolen or, if policy prohibits it, shared accounts.

> One method of looking for improperly used accounts is for the computer to inform users when they last logged on. This allows users to check if someone else used their account.

The secure storage of authentication data is also very important. This was discussed some in regard to password cracking. The confidentiality, integrity, and availability of authentication data is important. If the confidentiality is compromised, someone may be able to use the information to masquerade as a legitimate user. If system administrators can read the authentication file, they, too, can masquerade as another user. Many systems use encryption to hide the authentication data even from the system administrators.[64] If the integrity is compromised, authentication data can be added or the system can be disrupted. If the availability is compromised, the system cannot authenticate users.

16.4.2 Maintaining Authentication

So far, this chapter has discussed the initial authentication only. It is possible for someone to use someone else's account after the legitimate user has signed on.[65] Many computer systems handle this problem by logging a user out or locking their display or session after a certain period of inactivity. However, these methods can impact productivity and the make the computer less user-friendly.

16.4.3 Single Login

It is desirable for users to authenticate themselves only once and then to be able to access a wide variety of applications and data available on local and remote systems even if those systems require users to authenticate themselves. This is known as *single login*. If the access is within the same host computer, then the use of a modern access control system (such as an access control list) should allow for a single login. If the access is across multiple platforms,

[64] Masquerading *by* system administrators cannot be prevented entirely. However, controls can be set up so that improper actions by the system administrator can be detected in audit records.

[65] After a user signs on, the computer treats all commands originating from the user's physical device (such as a PC or terminal) as being from that user.

then the issue is more complicated, as discussed below.

There are three main techniques which can provide single login across multiple computers: host-to-host authentication, user-to-host authentication, and authentication servers.

Under a *host-to-host* authentication approach, the users authenticate themselves once to a host computer. That computer then authenticates itself to other computers and vouches for the specific user. Host-to-host authentication can be done by passing an identification, a password, or by a challenge-response mechanism or other one-time password scheme. Under this approach, it is necessary for the computers to recognize each other and to trust each other.

A *user-to-host* authentication approach requires the user to login to each host computer. However, a smart token (such as a smart card) can contain all the authentication data and perform the authentication for the user. To the user, it looks as though they were only authenticated once.

When using *authentication servers*, the users authenticate themselves to a special host computer (the authentication server). This computer then authenticates the user to other host computers the user wants to access. Under this approach, it is necessary for the computers to trust the authentication server.

> Kerberos and SPX are examples of network authentication servers. They both use cryptography to authenticate users to computers on the network.

16.5 Interdependencies

There are many interdependencies between identification & authentication and other controls. Several of them have been discussed in the chapter.

Logical Access Controls. Access controls are needed to protect the authentication database. Identification & authentication is often the basis for access controls. Dial-back modems and firewalls, discussed in the Access Control Chapter, can help prevent hackers from trying to login.

Audit. Identification & authentication is necessary for audit.

Physical Access Controls. Often the same methods used for authenticating a user to a computer system are used for physical access.

Personnel/User. Identification & authentication often applies to people. The administration of identification & authentication must be tied to personnel/user management to be effective.

Cryptography. Cryptography provides two basic services to identification & authentication: it protects the confidentiality of authentication data and it provides protocols for proving knowledge and/or possession of a token without having to transmit data which could be replayed to gain access to a computer system.

16.6 Cost Considerations

In general, passwords are the least expensive authentication technique and generally the least secure. They are already embedded in many systems. Memory tokens are less expensive than smart tokens, but have less functionality. Human interface smart tokens do not require readers, but are more inconvenient to use. Biometrics tend to be the most expensive.

For identification and authentication systems, the cost of administration is often underestimated. Just because a system comes with a password system, does not mean the system is free. For example, there is significant overhead to administering the I&A system.

16.7 Summary

Identification and authentication serves as a fundamental tool for securing computer systems and networks. It is used to identify users of automated resources and verify their identity. I&A is used for subsequent auditing of user activities and also is used as the basis for granting access to system resources. Technical implementations range from passwords, a low-cost and less secure measure, through biometrics, which, at relatively high-cost is best suited for situations requiring a very high degree of security.

16.8 References

Alexander, M. (editor) "Keeping the Bad Guys Off-Line." *Infosecurity News.* 4(6), 1993. pp. 54-65.

American National Standard for Financial Institution Sign-On Authentication for Wholesale Financial Transactions. ANSI X9.26-1990. American Bankers Association, Washington, D.C., February 28, 1990.

Automated Password Generator. FIPS PUB 181. National Institute of Standards and Technology. Octorer, 1993.

CCITT Recommendation X.509. The Directory - Authentication Framework. November 1988 (Developed in collaboration, and technically aligned, with 150 9594-8).

Feldmeier, David C. and Philip R. Kam, "UNIX Password Security - Ten Years Later." *Crypto '89 Abstracts*. Santa Barbara, CA, August 20-24, 1989.

Guideline for the Use of Advanced Authentication Technology. Draft FIPS PUB XXX.

National Institute of Standards and Technology.

Guideline on User Authentication Techniques for Computer Network Access Control. FIPS PUB 83. National Bureau of Standards, September 29, 1980.

Guidelines on Evaluation of Techniques for Automated Personal Identification. FIPS PUB 48. National Bureau of Standards, April 1, 1977.

Haykin, Martha E., and Robert B. J. Warnar. *Smart Card Technology: New Methods for Computer Access Control.* Spec. Pub. 500-157. National Institute of Standards and Technology, September 1988.

Kay, R. "Whatever Happened to Biometrics?" *Infosecurity News*. 4(5), 1993. pp. 60-62.

Morris, R. and K. Thompson. "Password Security: A Case History." *Communications of the ACM*, Vol. 22, No. 11, November 1979, pp. 594-597.

Needham, R. M. and M. D. Schroeder. "Using Encryption for Authentication in Large Networks of Computers." *Communications of the ACM*, Vol. 21, No. 12, December 1978, pp. 993-999.

Password Management Guideline. CSC-STD-002-85. Department of Defense. April 12, 1985.

Password Usage. FIPS PUB 112. National Bureau of Standards, May 30, 1985.

Salamone, S. "Internetwork Security: Unsafe at Any Node?" *Data Communications.* 22(12), 1993. pp. 61-68.

Sherman, R. "Biometric Futures." *Computers and Security*. 11(2), 1992. pp. 128-133.

Smid, Miles, James Dray and Robert B. J. Warnar. "A Token Based Access Control System for Computer Networks." *Proceedings 12th National Commuter Security Conference*. National Institute of Standards and Technology, October 1989.

Steiner, J.O., Neuman, C., and Schiller, J.1. "Kerberos: An Authentication Service for Open Network Systems." *Proceedings Winter USENIX*. Dallas, Texas, February 1988, pp. 191-202.

Troy, Eugene F. *Security for Dial-Up Lines*. Spec. Pub. 500-137, National Bureau of Standards, May 1986.

17 Logical Access Control

17.1 Introduction

> Logical Access Controls provide a technical means to control user access to information and system resources. This allows control over what information users can utilize, the programs they can run, and the modifications they can make.

On many multiuser systems, requirements to use (and prohibitions against use of) various computer resources[66] vary considerably. Typically, for example, some information must be accessible to all users[67], some may be needed by several groups or departments, and some should be accessed by only a few individuals. While it is obvious that users must have access to information that they need to accomplish their job, it may also be required to deny non-job-related access when not germane to assigned tasks. It may also be important to control the *kind of access* which is afforded (e.g. the ability for the average user to execute, but not change, system programs). This restriction of access helps to ensure that unauthorized actions are not taken.

> The term *access* is often confused with *authorization* and *authentication*.
>
> *Access* is the *ability* to do something with a computer resource. This usually refers to a technical ability (e.g., read, create, modify, or delete a file, execute a program, or use an external connection).
>
> *Authorization* is the *permission* to use a computer resource. Permission is granted, directly or indirectly, by the application or system owner.
>
> *Authentication* is proving (to some reasonable degree) that users are who they claim to be.

Access is the ability to do something with a computer resource (e.g., use, change, or view). *Access control* is the means by which the ability is explicitly enabled or restricted in some way (usually through either physical or system-based controls). *Logical* access control is the technical means of restricting who or what can use system resources. Logical access controls can prescribe not only who or what (e.g., in the case of a process) is to have access to a particular system resources but also the type of access that is permitted. These controls may be built into the operating system, may be incorporated into applications programs or major utilities (e.g., database management systems), or may also be implemented in add-on security packages that are installed into an operating system. Additionally, access controls may be present in specialized components that regulate communications between computers.

---

[66] In this chapter, use of the term *resources* includes information as well as system resources, such as programs, subroutines, hardware (e.g., modems, communications lines) and so forth.

[67] Note that "users" need not be actual human users. They could include, for example, a program or another computer requesting use of a system resource.

Logical access controls can help protect:

- operating systems and other system software from unauthorized modification or manipulation (and thereby help ensure system integrity and availability);
- the integrity and availability of information by restricting the number of users and processes with access; and
- confidential information from being disclosed to unauthorized individuals.

Controlling access is normally thought of as applying to human users (e.g., will technical access be provided for user JSMITH to the file "payroll.dat") but *this is not always the case*. Access can be provided to other computer systems. Also, access control is often incorrectly thought of as only applying to *files*. However, it also protects other system resources such as the ability to place an outgoing long distance phone call though a system modem (as well as, perhaps, the information that can be sent over such a call). Access controls can also apply to specific functions within an application and in restricting access to specific fields of a file.

This chapter first discusses basic criteria that can be used to decide whether a particular user should be granted access to a particular system resource. The use of these criteria by those that set policy (usually system-specific policy) is then briefly reviewed. Commonly used *technical mechanisms* for implementing logical access control are then presented, followed by a brief discussion of issues related to access control administration. Finally, cost considerations of logical access control implementation and administration and interdependencies with other security controls are examined.

17.2 Access Criteria

When making a determination about what kind of technical access to allow to some data, programs, devices, and resources, it is important to consider who will have access and what kind of access will they be allowed. It may be desirable that some information on the system, such as the data displayed on an organization's daily calendar of non-confidential meetings, be readable by everyone in the organization. The program that formats and displays the calendar, however, might be modifiable by only a very few system administrators, while the operating system controlling that program might be directly accessible by still fewer.

In deciding whether to permit use of a system resource by a particular user, logical access controls examines the question: **Is the requestor authorized to the type of access requested?** (Note that this is usually distinct from the question of whether the user is authorized to use the system *at all*, which is usually addressed in an identification and authentication process.)

This section examines the various criteria that the system uses to determine if a request for access will be granted. (The technical mechanisms for implementing these decisions are discussed below.) They are typically used in some combination. Many of the advantages as well as many of the complexities involved in implementing and managing access control are related to the different kinds of user accesses supported.

17.2.1 Identity

It is probably fair to say that the majority of access controls are based upon the identity of the user (either human or process), which is usually obtained through identification and authentication (I&A). (See the I&A chapter.) The identity is usually unique but can be a group identification or even anonymous. For example, public information dissemination systems may serve a large group called "researchers" in which the individual researchers are not known.

17.2.2 Roles

Access to information may also be controlled by the job assignment or function (i.e., the *role*) of the user that is seeking access. Examples of roles include data entry clerk, purchase officer, project leader, programmer, technical editor, and so forth. Access rights are grouped by role name and use of resources is restricted to individuals authorized to assume the associated role. An individual may be authorized for more than one role, but may be required to act in a single role at a time. Changing roles may require logging out and then in again, or entering a role-changing command. Note that use of roles is *not* the same as shared-use accounts. An individual may be assigned the rights of a shipping department data entry clerk, for example, but the account would still be tied to that individual's identity to allow for auditing. (See the Audit chapter.)

> Many systems already support a small number of special purpose roles, such as System Administrator or Operator. An individual who is logged on in the role of a System Administrator can, for example, perform operations that would be denied to the same individual acting in the role of an ordinary user. Recently, the use of roles has been expanded beyond system tasks to application oriented activities. For example, in a company there could be an Order Taking Role. A user with this role would be able to collect and enter customer billing information, check on availability of particular items, request shipment of items, and issue invoices. In addition, there could be an Accounts Receivable Role which would receive payments and credit them to particular invoices. A third, Shipping Role, could then be responsible for shipping products and updating the inventory. To provide additional security, constraints could be imposed such that a single individual user would never be simultaneously authorized to assume all three roles. Constraints of this kind are sometimes referred to as separation of duty constraints.

The use of roles and the functions that each requires can be a very effective way of providing access control. The process of defining roles should be based on a thorough analysis of the way in which an organization operates and should include input from a wide spectrum of users in an organization.

17.2.3 Location

Access to particular system resources may also be based upon physical or logical location. For example, in a prison, all users in areas to which prisoners are physically permitted may be limited to read-only access. Changing or deleting is limited to areas to which prisoners are denied physical access. The same authorized users (e.g., prison guards) would operate under significantly different logical access controls depending upon their physical location. Similarly, users can be restricted based upon network addresses (e.g., users from sites within a given organization may be permitted greater access than those from outside).

17.2.4 Time

Time of day or day of week restrictions are also commonly utilized limitations on access. For example, use of confidential personnel files may be allowed only during normal working hours -- and denied before 8:00 a.m. and after 6:00 p.m. and all day during non-work days (Saturdays, Sundays and holidays).

17.2.5 Service Constraints

Service constraints are those restrictions put in place that depend upon the parameters that may arise during use of the application or are pre-established by the resource owner/manager. For example, a particular software package may only be licensed by the organization for five users at a time. Access would be denied for a sixth user, even if the user were otherwise authorized to use the application. Another type of service constraint is based upon application content or numerical thresholds. For example, an ATM machine may restrict transfers of money between accounts to certain dollar limits or limiting maximum ATM withdrawals to $500 per day. Access may also be selectively permitted based on the type of service requested. For example, users of computers on a network may be permitted to exchange electronic mail, but might not be allowed to login to each others' computers.

17.2.6 Common Access Modes

These are used to permit or deny specific modes or types of access *by specific users* (individuals, groups, or other systems) to specific system resources. They are often used to restrict access to a wide variety of system resources such as files, specific records, fields, output or storage devices (e.g., on a large multiuser system, user files may be maintained on separate media from system files).

The concept of access modes is fundamental to access control. A brief overview of common access modes, which can be used in both operating or application systems, follows:[68]

> *Read* access provides users with the capability to view, copy, and usually print the information in a system resource (such as a file, certain records, certain fields, or some combination thereof), but not to *alter* it, such as delete from, add to, or modify it in

[68] These access modes are described generically; exact definitions and capabilities will vary from implementation to implementation. Readers are advised to consult their system and application documentation.

any way. One must assume that information can be copied (although perhaps only manually) if it can be read.

Write access allows users to add to, modify, or delete information in system resources (e.g., files, records, programs). Normally a user has read access to anything they have write access to.

Execute privilege allows users to run programs.

Delete access allows users to erase system resources (e.g., files, records, fields, programs).[69] Note that if someone has write access but not delete access, they could overwrite the field or file with gibberish or otherwise inaccurate information and, in effect, delete the information.

Other specialized access modes (more often found in applications) include:

Create access allows users to create new files, records, or fields.

Search access allows users to list the files in a directory.

Of course, these criteria can be used in conjunction with one another. For example, an organization may give authorized individuals write access to an application at any time from within the office but only read access during normal working hours if they dial-in.

Depending upon the technical mechanisms available to implement logical access control, a wide variety of access permissions and restrictions are possible. No discussion can present all possibilities.

17.3 Policy: the Impetus for Access Controls

> This chapter discusses issues relating to the technical implementation of logical access controls -- not the actual policy decisions as to who *should* have what type of access. These decisions are typically included in system-specific policy, as discussed in the policy and personnel chapters.

Logical access controls are a technical means of implementing *policy decisions*. Policy is made by a management official responsible for a particular system, application, or group of systems. The development of an access control policy for an application, system, or group of systems (among other possibilities) is not an easy endeavor. It is necessary to balance the often competing interests of, but not limited to:

[69] Note that "deleting" information does not necessarily physically remove the data from the storage media. This can have serious implications for information which must be kept confidential. See CSL Bulletin, *Disposition of Sensitive Automated Information*, NIST, October 1992.

security, operational requirements, and user friendliness. In addition, technical constraints must be considered. A few *simple* examples of *specific policy issues* are provided below; it is important to recognize, however, that *comprehensive system-specific policy* is significantly more complex.

For example, the director of an organization's personnel office would normally decide who would have access to the personnel application(s) and the kinds of access they should have.

As another example, in a disbursing office, a single individual is usually prohibited from both requesting and authorizing that a particular payment be made. This is a *policy decision* taken to reduce the likelihood of embezzlement and fraud. This policy can be implemented (or *enforced*) on systems through the use of logical access controls -- in both the operating system and disbursing application.

Decisions may also be made regarding access to the system itself. For example, the senior IRM official may decide that agency systems that process information protected by the Privacy Act may not be used to process public-access database applications.

Once these policy decisions have been made, they will be *implemented* through logical access controls. In doing so, it is important to realize that the capabilities of various types of technical mechanisms (for logical access control) vary greatly.[70]

17.4 Technical Implementation Mechanisms

Many mechanisms have been developed to provide access control on systems, and they vary significantly in terms of precision, sophistication, and cost. This section will provide an overview of some of the methods. It should be noted that these methods are not mutually exclusive and are often employed in combination. Managers need to analyze their organization's protection requirements in selecting the most appropriate, cost-effective access control method(s).

This section contains two subsections. The first addresses *internal* access control mechanisms while the second deals with access control mechanisms for *external* connections.

17.4.1 Internal Access Controls

Internal access controls are a logical means of separating what defined users (or user groups) can or cannot do with system resources.

Four methods of internal access control are discussed in this section: Passwords, access control lists, constrained user interfaces, and labels.

---

[70] Some policies may not be technically implementable; appropriate technical controls may simply not exist.

17.4.1.1 Passwords and Other I&A Mechanisms

> The use of passwords as a means of access control can result in a proliferation of passwords that can reduce overall security.

Passwords are most often associated with technical user authentication. (See the I&A chapter.) However, they are also used to protect data and applications on many systems, including PCs. For instance, an accounting application may require a password in order to access certain financial data or to invoke a restricted application (or function of an application).[71]

Password-based access control is often inexpensive because it is already included in a large variety of applications. However, users may find it difficult to remember additional application passwords which, if written down or poorly chosen, can lead to their compromise. PC password-based access controls may be easy to circumvent. As discussed in the I&A chapter, there are many disadvantages to using passwords.

Another mechanism that can be used for logical access control is cryptography. Encrypted information can only be decrypted by those possessing the appropriate cryptographic key. This is especially useful if physical access cannot be ensured, such as for laptops or floppy diskettes. Thus, for example, if a laptop computer is stolen, the information can not be accessed. While cryptography can provide strong access control, it is accompanied by the need for strong key management administration and may affect availability (e.g., if keys are lost or stolen). In addition, minor read/write errors may prevent the decryption of the information. (See the cryptography chapter.)

> **System and Security Administration**
>
> In order to conduct system and security administration, access to advanced functions (such as setting up a user account) is necessary. The individuals that technically setup and modify who has access to what are very powerful users on the system. Such users are often called system or security administrators. On some systems, these are referred to as *privileged accounts*.
>
> The type of access of these accounts varies considerably. Some administrator privileges, for example, may allow an individual to administer only one application or subsystem, while a higher level of privileges may allow for oversight and establishment of subsystem administrators.
>
> Normally, users who are security administrators have two accounts: one for regular use and one for security use. This can help protect the security account from compromise. Furthermore, additional I&A precautions, such as ensuring that administrator passwords are robust and changed regularly, are important to minimize opportunities for unauthorized individuals to gain access to these functions.

[71] Note that this password is normally *in addition* to the one supplied initially to log onto the system.

17.4.1.2 Access Control Lists

Broadly speaking, an Access Control List (ACL) refers to a register of: 1) those users (including groups, machines, processes) who have been given the ability to use a particular system resource, and 2) the types of access they have been permitted.

ACLs vary considerably in their capability and flexibility. Some only allow specification for certain pre-set groups (e.g., owner, group, and world) while more advanced ACLs allow much more flexibility, such as *user defined* groups. Also, more advanced ACLs can be used to explicitly *deny* access to a particular individual or group.[72] When using more advanced ACLs, access can be at the discretion of the policy maker (and implemented by the security administrator) or individual users, depending upon how the controls are technically implemented.

Elementary ACLs

Elementary ACLs (e.g., "permission bits") are a widely available means of providing access control on multi-user systems. In this scheme, a short, pre-defined list is maintained of the access rights to files or other system resources.

> Elementary ACLs typically assign read, write, execute or delete capabilities to: 1) the resource owner, 2) the group to which the resource is related; and 3) everyone else (the "world").

Elementary ACLs are typically based on the concepts of *owner, group, and world*. For each of these, a set of access modes (typically chosen from read, write, execute, and delete) is specified by the owner (or custodian) of the resource. The owner is usually its creator, though in some cases application systems may be automatically assign ownership of all resources system to project administrators regardless of the identity of their creator. Owners often have all privileges for their resources.

In addition to the privileges assigned to the owner, each resource is also associated with *a named group of users*. Users who are members of the group can be granted modes of access distinct from non-members, who belong to the rest of the "world" that includes all of the system's users. User groups may be arranged according to departments, projects, or ways

[72] Since one would presume that no one would have access without being granted access, why would it be desirable to explicitly deny access? Consider a situation in which a group name has already been established for, say fifty, employees. If it were desired to exclude five of the individuals from that group, it would be easier for the access control administrator to simply grant access to that group and take it away from the five rather than grant access to 45 people. Or, consider the case of a complex application in which many groups of users are defined. In may be desired, for some reason, to prohibit Ms. *X* from generating a particular report (perhaps she is under investigation). In a situation in which group names are used (and perhaps modified by others), this explicit denial may be a safety check to restrict Ms. *X*'s access -- in case someone were to redefine a group (with access to the report generation function) to include Ms. *X*. She would still be denied access.

appropriate for the particular organization. For example, groups may be established for members of the Personnel and Accounting departments. The system administrator is normally responsible for technically maintaining and changing the membership of a group, based upon input from the owners/custodians of the particular resources to which the groups may be granted access.

Example of Elementary ACL for the file "payroll":

Owner: PAYMANAGER
Access: Read, Write, Execute, Delete

Group: COMPENSATION-OFFICE
Access: Read, Write, Execute, Delete

"World"
Access: None

As the name implies, however, the technology is not particularly flexible. It is not possible to explicitly deny access to an individual that is a member of the file's group. Also, it is not possible for two groups to easily share information (without exposing it to the "world"), since the list is pre-defined to only include one group. If two groups wished to share information, an owner may make the file available to be read by "world." This may disclose information that should be restricted. Unfortunately, elementary ACLs have no mechanism to easily permit such sharing.

Advanced ACLs

Like elementary ACLs, advanced ACLs provide a form of access control based upon a logical registry. They do, however, provide *finer precision* in control.

Example of Advanced ACL for the file "payroll"

PAYMGR:	R,	W,	E,	D
J. Anderson:	R,	W,	E,	-
L. Carnahan:	-,	-,	-,	-
B. Guttman:	R,	W,	E,	-
E. Roback:	R,	W,	E,	-
H. Smith:	R,	-,	-,	-
PAY-OFFICE:	R,	-,	-,	-
WORLD:	-,	-,	-,	-

Advanced ACLs provide a fine grained form of access control that can be very useful in many complex information sharing situations. Their flexibility also makes them more of a challenge to manage. The rules for determining access in the face of apparently conflicting ACL entries are not uniform across all implementations and can be confusing to security administrators. When such systems are introduced, they should be coupled with training to ensure their correct use.

Advanced ACLs, however, provide a great deal of flexibility in implementing system-specific policy. They allow for easy customization to meet the security requirements of functional managers.

17.4.1.3 Constrained User Interfaces

Often used in conjunction with ACLs are *constrained user interfaces* which restrict users' access to specific functions by never allowing them to request the use of information, functions or other specific system resources for which they do not have access. Three major types exist: 1) *menus*, 2) *database views*, and 3) *physically constrained user interfaces*.

Data Categorization

One tool that is often used to increase the ease of security labelling is by categorizing data by similar protection requirements. For example, "organization proprietary data" may be disclosable only to the organization's employees while "public data" may be available to anyone.

Constrained user interfaces can provide a form of access control that closely models the way in which an organization operates. Many systems allow administrators to restrict users ability to use the operating system or application system directly. Users can only execute commands that are provided by the administrator, typically in the form of a menu. Another means of restricting users is through restricted *shells* which limit the system commands the user can invoke. The use of menus and shells can often make the system easier for non-technical users and reduce errors.

Menu driven systems are a common constrained user interface, where different users are provided different menus on the same system. A user is not given menu options for unauthorized operations and so has no means by which to invoke them.

Database views are a mechanism for restricting user access to data contained in a database. It may be necessary to allow a user to access a database, but that user might not need access to all data in the database (e.g., not all fields of a record nor all records in the database). Views can be used to enforce complex access requirements that are often needed in database situations, such as those based on the content of a field. For example, consider the situation where clerks maintain personnel records in a database. Clerks are assigned a range of clients based upon last name (e.g., A-C, D-G, etc.). Instead of granting a user access to all records, the view can grant the user access to the record based upon the first letter of the last name field.

Use of *physically* constrained user interfaces can also provide limitation of users' abilities. A common example is an ATM machine, which provides only a limited number of physical buttons to select options; no alphabetic keyboard is usually present.

17.4.1.4 Security Labels

A security label is a designation assigned to each *file* (and sometimes to parts of files and also to other system resources) that indicates the applicable protection requirements of that data (e.g., "PERSONAL" may be a label applied to data covered by the Privacy Act). Labels can be used for a variety of purposes including access control, specifying protective measures, and indicating additional handling instructions. In many implementations, once this designator has been set, it cannot be changed (except perhaps under carefully controlled conditions that are subject to auditing).

When used for access control, labels are also assigned to *user sessions* that designate the kinds of data that can be used during the session by the particular user. In systems implementing labels, users are permitted to initiate sessions with specific labels only. For example, a file bearing the label "Organization Proprietary Information" would not be accessible (readable) except during user sessions with the corresponding label. Moreover, only a restricted set of users would be able to initiate such sessions. The labels of the session and those of the files accessed during the session are used, in turn, to label output from the session. This ensures that information is uniformly protected throughout its life on the system.

For systems with stringent security requirements (such as those processing national security information) labels may be useful in access control.

Labels are a very strong form of access control; however, they are often inflexible and can be expensive to administer. Unlike permission bits or access control lists, labels cannot ordinarily be changed. Since labels are permanently linked to specific information, data cannot be disclosed by a user copying information and changing the access to that file so that the information is more accessible than the original owner intended. By removing users' ability to arbitrarily designate the accessibility of files they own, opportunities for certain kinds of human errors and malicious software problems are eliminated. In the example above, it would not be possible to copy Organization Proprietary Information into a file with a different label. This prevents inappropriate disclosure, but it may also interfere with legitimate extraction of some information.

Labels are well-suited for consistently and uniformly enforcing access restrictions, although their administration and inflexibility can be a significant deterrent to their use.

17.4.2 External Access Controls

External access controls are a means of controlling interactions between the system and outside people, systems, and services. External access controls use a wide variety of methods often including separate physical devices

17.4.2.1 Port Protection Devices

A *port protection device* (PPD) is an external device fitted to a communications port of a host computer that provides the function of authorizing access to the port itself, prior to and independent of the computer's own access control functions. A PPD can be a separate device in the communications stream[73] or it may be incorporated into a communications device (e.g., a modem). PPDs typically require a separate authenticator, such as a password, in order to

[73] Typically PPDs are found only in serial communications streams.

access the communications port.

One of the most common type of PPD is the *dial-back modem*. A typical dial-back modem sequence follows: a user calls the dial-back modem and enters a password. The modem hangs up on the user and performs a table lookup for the password provided. If the password is found, the modem places a return call to the user (at a previously specified number) to initiate the session. The return call itself also helps to protect against the use of lost or compromised accounts.[74]

17.4.2.2 Secure Gateways

Secure gateways, often called *firewalls*, block or filter access between two networks, often between a private[75] network and a larger more public network such as the Internet. These larger, more public networks, unfortunately, contain hackers. The purpose of secure gateways is to allow internal users to benefit from connecting to external networks while not allowing hackers to compromise internal systems.[76]

> There are many types of secure gateways. Some of the most common are packet filtering (or screening) routers, proxy hosts, bastion hosts, dual-homed gateways, screened host gateways.

In order to use a gateway, network administrators funnel all traffic between the two networks through one point, the gateway. The gateway then allows or disallows the traffic to pass through. Some gateways are set up to allow all traffic except for specific traffic which has known or suspected vulnerabilities or security problems, such as remote login services. Other gateways disallow all traffic except for specific types, such as e-mail. Some gateways can make access control decisions based on the location of the requester. There are several technical approaches and mechanisms used to support secure gateways.

In general, the stronger the security, the more restrictive the gateway. For this reason, firewall experts always emphasize the need for policy so that appropriate officials make the decision where the organization's tradeoff should be between operational needs and security.

---

[74] This is, however, not always the case. Clever malicious hackers in the phone system can use such advanced functions as call forwarding to re-route calls.

[75] Private networks is somewhat of a misnomer. Firewalls protect networks that an organization desires to restrict, in some way(s) from use by outsiders. It does not mean that the organization's network is totally inaccessible to outsiders or prohibits use of the outside network from insiders (or the network would be disconnected). It also does not mean that all the information on the network requires confidentiality protection.

[76] Questions frequently arise as to whether secure gateways help prevent the spread of viruses. Since viruses are transmitted as a part of a program, gateways *cannot* help prevent their propagation. However, they may aid the spread of network worms.

In addition to reducing the risk of hackers, there are other benefits to using secure gateways. Gateways can reduce internal system security overhead since they allow an organization to concentrate security efforts on a limited number of machines. (This is similar to putting a guard on the first floor of a building instead of needing a guard on every floor.) Instead of trying to be sure that every machine on a private network is operating at a high level of security (e.g., has all pertinent bug fixes and security patches installed in a timely manner, which is improbable at best), a secure gateway allows an organization to focus their security efforts on maintaining relatively few machines at a high level of security.

A second benefit is the centralization of services. A secure gateway can be used to provide a central management point for various services, such as advanced authentication (discussed in the I&A chapter), e-mail, or public dissemination of information. Having a central management point can reduce system overhead and improve service.

17.4.2.3 Host-based Authentication

> An example of host-based authentication would be the Network File System (NFS). NFS allows a server to make filesystems/directories available to specific machines.

Host-based authentication grants access based upon the *identity of the host* originating the request instead of the identity of the user making the request. Many network applications in use today utilize host-based authentication to determine whether or not access is allowed. Under certain circumstances it is fairly easy to masquerade as the legitimate host, especially if the masquerading host is physically located close to the host being impersonated. Security measures to protect against misuse of some host-based authentication systems are available (e.g., Secure RPC[77] uses DES to provide a more secure identification of the client host).

17.5 Administration of Access Controls

Administration is the most complex and challenging aspect of access control. Administration of access controls involves implementing, monitoring, modifying, testing, and terminating user accesses on the system. These can be demanding tasks, even though they typically do not include making the actual decisions as to the type of access each user may have. Those decisions are usually the responsibility of the applicable application manager or cognizant management official.[78] Decisions regarding accesses should be guided by organizational policy, employee job descriptions and tasks, information sensitivity, user "need to know" determinations, and many other factors.

---

[77] RPC, or Remote Procedure Call, is the service used to implement NFS.

[78] As discussed in the policy section earlier in this chapter. See also the discussion of system-specific policy in both the Policy and Personnel chapters.

Implementation and management of logical access controls are accomplished through an administrative function. There are three basic approaches to administering access controls: centralized, decentralized, or a combination. Each has relative advantages and disadvantages, and which is most appropriate in a given situation will depend upon the particular organization and its circumstances.

17.5.1 Centralized Administration

Using centralized administration, one office or individual is responsible for configuring access controls. As users' information processing needs change, their accesses can be modified only through the central office, usually after requests have been approved by the appropriate official. This allows very strict control over information to be maintained because the ability to make changes resides with very few individuals. Each user's account can be centrally monitored, and closing all accesses for any user can be easily accomplished if that individual leaves the organization. Since relatively few individuals oversee the process, consistent and uniform procedures and criteria are usually not difficult to enforce. However, when changes are needed quickly in order for users to complete important tasks, going through central administration can be frustrating and time-consuming.

17.5.2 Decentralized Administration

In decentralized administration, access is directly controlled by the owners or creators of the files, whoever or wherever those individuals may be. This keeps control in the hands of those most accountable for the information, most familiar with it and its uses, and best able to judge who needs access and of what type(s). This may lead, however, to a lack of consistency among owners/creators as to procedures and criteria for granting user accesses and capabilities. Also, when requests are not processed centrally, it may be much more difficult to form a system-wide composite view of all user accesses on the system at any given time. Different data owners may inadvertently implement combinations of accesses that introduce conflicts of interest or that are in some other way not in the organization's best interest.[79] It may also be difficult to ensure that accesses are properly terminated when an employee transfers within or leaves an organization.

17.5.3 Hybrid Approach

In the hybrid approach, centralized control is exercised for some information and decentralized is allowed for other information. One typical arrangement is that central administration is responsible for the broadest and most basic accesses, and the owners/creators of files control types of accesses or changes in users' abilities for the files under their control. The main disadvantage to a hybrid approach is adequately defining which accesses should be assignable locally and which centrally.

---

[79] Note that, without necessary review mechanisms, central administration does not *a priori* preclude this.

17.6 Interdependencies

Policy - The most fundamental interdependency of logical access control is with policy. Logical access controls are the technical implementation of system-specific and organizational policy which specifies who should be able to access what kinds of information, applications, and functions.

Audit - As discussed earlier, logical access controls can be difficult to implement correctly. Also, it is sometimes *not possible* to make logical access control as precise, or fine-grained, as would be ideal for an organization. In such situations, users may, either deliberately or inadvertently, abuse their access. For example, access controls cannot prevent a user from modifying data the user is authorized to modify, even if the modification is incorrect. Auditing provides a way to identify abuse of access permissions. It also provides a means to audit the actions of system or security administrators.

Identification & Authentication - In most logical access control scenarios, the identity of the user must be established before an access control decision can be made. The access control process then associates the appropriate information and permissible forms of accesses with that identity. This means that access control can only be as effective as the I&A process employed for the system. If users tell one another passwords or write them down, both I&A and access control for the system are compromised.

Physical Access Control - Logical access control can enhance the security provided by physical access control by acting as an additional guard against unauthorized access to, or use of, the system's resources. It can also augment physical access control by providing added precision, since different users are able to perform different functions. However, most systems can be compromised if someone can physically access the machine (i.e., CPU or other major components) by, for example, restarting the system with different software.

17.7 Cost Considerations

Incorporating logical access control into a computer system involves both the purchase or utilization of access control mechanisms, their implementation, as well as a change in behavior on the part of users.

Direct Costs - Among the direct costs associated with the use of logical access control methods are the purchase and support of hardware, operating systems and applications that provide the controls, and any necessary add-on security packages. The most significant personnel cost in relation to logical access control is usually for administration (e.g., initially determining, assigning and keeping access rights up to date). Label-based access control is available in a limited number of commercial products, but at greater cost and with less variety of selection. Role-based systems are becoming more available, but there are significant costs involved in customizing these systems for a particular organization. Training users to understand and use an access control system is another necessary cost.

Indirect Costs - The primary indirect cost associated with introducing logical access controls into a computer system is the effect on user productivity. There may be additional overhead involved in having individual users properly determining (when under their control) the protection attributes of information. Another indirect cost that may arise results from users not being able to immediately access information necessary to accomplish their jobs because the permissions were incorrectly assigned (or have changed). This situation is familiar to most organizations that put strong emphasis on logical access control.

17.8 Summary

Logical access controls can limit access to information and restrict the capabilities or modes of access users have; however, they are not cost-free and their implementation does not automatically make a system more secure. Poorly chosen or improperly configured control mechanisms can be detrimental, as can inadequate understanding of the complexities involved in implementing and managing this technology. Many technical means exist for organizations to implement access control, with widely varying acquisition and administrative costs.

17.9 References

Abrams, M.D., et al. *A Generalized Framework for Access Control: an Informal Description*. McLean, VA: Mitre Corporation, 1990.

Bach, M.J. *The Design of the Unix Operating System*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

Baldwin, R.W. "Naming and Grouping Privileges to Simplify Security Management in Large Databases." *1990 IEEE Symposium on Security and Privacy Proceedings*. Oakland, CA: IEEE, May 1990, pp. 116-132.

Boebert, W. E. and R. Y. Kain. "A Practical Alternative to Hierarchical Integrity Policies." *8th National Computer Security Conference Proceedings*. National Institute of Standards and Technology, September 1985, pp. 18-27.

Caelli, William, et al. *Information Security Handbook*. New York, NY: Stockton Press, 1991.

Cheswick, William and Seven Bellovin. *Firewalls and Internet Security*. Addison-Wesley Publishing Compang 1194.

Clark, D. and D. Wilson. "A Comparison of Commercial and Military Computer Security Policies." *1987 IEEE Symposium on Security and Privacy Proceedings*. Oakland, CA: IEEE, April 1987, pp. 184-194.

Curry, D. *Improving the Security of Your UNIX System, ITSTD-721-FR-90-21.* Menlo Park, CA: SRI International, 1990.

Dinkel, Charles. *Secure Data Network System Access Control Documents*. NISTIR 90-4259. National Institute of Standards and Technology, 1990.

Garfinkel, S. and Spafford, G. "UNIX Security Checklist." *Practical UNIX Security*. Sebastopol,CA: O.Riley & Associates. Inc., 1991, pp. 401-413.

Gasser, Morrie. *Building a Secure Computer System*. New York, NY: Van Nostrand Reinhold Company, Inc., 1988.

Guideline on User Authentication Techniques for Computer Network Access Control. FIPS PUB 83. National Institute of Standards and Technology, September 1980.

Guidelines for Security of Computer Applications. FIPS PUB 73. National Institute of Standards and Technology, June 1980.

Haykin, M. and R. Warner, *Smart Card Technology: New Methods for Computer Access Control*. Spec Pub 500-157. National Institute of Standards and Technology, 1988.

Landwehr, C., C. Heitmeyer and J. McLean. "A Security Model for Military Message Systems." *ACM Transactions on Computer Systems*, Vol 2, No.3, August 1984.

Pfleeger, Charles P. *Security In Computing*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1989.

S. Salamone, "Internetwork Security: Unsafe at Any Node?" *Data Communications*, 22(12), 1993, pp. 61-68.

Sandhu, R. "Transaction Control Expressions for Separation of Duty." *Fourth Annual Computer Security Applications Conference Proceedings*. Orlando, FL, December 1988, pp. 282-286.

Thomsen, D.J. "Role-based Application Design and Enforcement." *Fourth IFIP Workshop on Database Security Proceedings*, Halifax, England, September 1990.

T. Whiting, "Understanding VAX/VMS Security." *Computers and Security*, 11(8), 1992, pp. 695-698.

18 Audit Trails

18.1 Introduction

> **Definition**
>
> An *audit trail* can be thought of as a series of records of computer events, about an operating system, an application, or user activities. A computer system may have several audit trails, each devoted to a particular type of activity.
>
> More formally, an audit trail is defined as "a chronological record of system activities which is sufficient to enable the reconstruction, review, and examination of the sequence of environments and activities surrounding or leading to each event in the path of a transaction from its inception to output of final results." [FIPS 39]

Audit trails maintain a record of system activity, by both system and application processes, and activity by users of systems and applications.[80] In conjunction with appropriate tools and procedures, audit trails can assist an administrator in detecting security violations, performance problems, and flaws in applications. The process of *auditing* can be a powerful security tool.

Audit trails may be used in two ways: as either a kind of insurance policy or as a procedure in regular system operations (or both). As insurance, audit trails are maintained but are not used unless needed. As a procedure, auditing is used to help system administrators ensure that the system or resources have not been harmed by hackers, insiders, or technical problems.

Overview of the Chapter

This chapter is divided into five sections:

- Benefits and Objectives;
- Audit Trail and Logs;
- Audit Trail Management Issues;
- Interdependencies; and
- Cost Considerations.

18.2 Benefits and Objectives

Auditing can provided a means to help accomplish *several* security related objectives, including:

- Individual accountability;

[80] The type and amount of detail recorded by audit trails varies by both the technical capability of the logging application and managerial decisions. Therefore, when we state that "audit trails can..." the reader should be aware that not all audit trails may have this capability.

- Reconstruction of events;
- Intrusion detection; and
- Problem analysis.

An *event* is any action that happens on a computer system. Examples include logging into a system, execution of a program, and opening a file.

Auditing can help to provide for individual accountability by providing a record of actions performed by users. If users are aware that their activities are being recorded (and can therefore presumably be traced back to them), they are less likely to perform unauthorized or inappropriate activities. This record of actions can also be used to *reconstruct events* to assist in recovery efforts, usually from technical accidents. Recognition of specific sequences of events, such as those indicative of unauthorized access, can aid in *intrusion detection*. Finally, analysis of audit records can be a useful tool in *problem analysis* by showing the source of anomalous behavior in both the system and applications. Each is discussed below.

18.2.1 Individual Accountability

Audit trails are a technical mechanism that assists management to help ensure individual accountability of system users. By advising users that they are personally accountable for their actions, which are tracked by an audit trail that logs user activities, management can help promote proper user behavior.[81] Users are less likely to attempt to circumvent security policy if they know their name will show up in an audit log.

For example, audit trails can be used in concert with access controls to identify and provide information about users suspected of improper modification of data (e.g., introducing errors into a database). An audit trail may be used to monitor and log "before" and "after" versions of records. (Depending upon the size of the file and the capabilities of the audit logging tools, this may be very resource-intensive.) Comparisons can then be made between the actual changes made to records and what was expected. This can help management determine whether errors were made by the user or were caused by system or application software errors, or some other non-user error.

Audit trails work in concert with logical access controls, which restrict use of system resources -- both functions and information. Granting a use access to a particular resource usually means that they need that access to accomplish their job. Authorized access, of course, can be misused, which is where auditing is useful. While users cannot be prevented from using resources to which they have legitimate access authorization, auditing is used to check upon their actions. For example, consider a personnel office in which users have access those personnel records for which they are responsible. Audit trails can reveal that a particular user is printing records far more than the average user, which could indicate the selling of personal data. Another example may be an engineer who is using a computer for

[81] For a fuller discussion of changing employee behavior, see the training chapter.

the design of a new product. Auditing could reveal that an outgoing modem was used extensively by the engineer the week before quitting. This could be used to investigate whether proprietary data files were sent to an unauthorized party.

Furthermore, audit trails can discourage sharing of passwords or poorly chosen passwords. Audit trails can track activity by user identification. By itself, the audit trail is generally *unable* to differentiate between a legitimate authorized user properly logged on and one who has borrowed or stolen a password and user identification (or defeated another I&A mechanism).[82] Therefore, users that weaken I&A mechanisms (e.g., by sharing passwords or leaving a token & PIN on their desk) could be blamed for someone else's actions.

18.2.2 Reconstruction of Events

Audit trails can also be used to reconstruct events after a problem has occurred. Damage assessment can be more easily accomplished by reviewing audit trails of system activity to pinpoint how, when, and why normal operations ceased. System audits can often be used to distinguish between operator-induced errors (during which the system may have performed exactly as intended) or system-created errors (e.g., arising from a poorly tested piece of replacement code). If, for example, a system fails or the integrity of a file (either program or data) is questioned, an audit of system activity can be used to reconstruct the series of steps taken by the system, users, and the application. Knowledge of the conditions that existed at the time of, for example, a system crash, can be useful in avoiding future crashes. Additionally, if a technical calamity occurs (e.g., the corruption of a data file) audit trails can aid in the recovery process (e.g., by using the record of changes made to reconstruct the file).

18.2.3 Intrusion Detection

> *Intrusion detection* refers to the process of identifying attempts to penetrate a system and gain unauthorized access.

If audit trails have been designed and implemented to record the appropriate information, they can assist in intrusion detection. Although normally thought of a real-time effort, *intrusion detection* can occur either *in real-time*, by examining audit records as they are created (or through the use of other kinds of warning flags/notices), or *after the fact* (e.g., by examining audit records in a batch process).

Real-time intrusion detection is primarily used to detect outsiders attempting to gain unauthorized access to the system. It may also be used to detect changes in system performance indicative of, for example, a virus or worm attack. There may be difficulties in implementing real-time auditing including unacceptable system performance or the purchase

[82] If used with artificial intelligence analysis or other techniques, this distinction can be identified. See the discussion on intrusion detection which follows in this chapter.

of adjunct processors.

After-the-fact identification may indicate that unauthorized access was attempted (or succeeded). Attention can then be given to damage assessment or reviewing controls that were attacked.

18.2.4 Problem Analysis

Audit trails may be used as on-line tools to help identify problems as they occur. This is often referred to as *real-time auditing* or monitoring. If a system or application is deemed to be critical to an organization's business or mission, real-time auditing may be implemented that can be used to monitor the status of these processes (although, as noted above, there can be difficulties with real-time analysis). System audits can be used to verify that the *system* operated normally (i.e., that an error may have resulted from operator error as opposed to a system-originated error). Such auditing may be complemented by system performance logs. For example, a significant increase in the use of system resources (e.g., disk file space or outgoing modem use) *could* indicate a security problem.

18.3 Audit Trails and Logs

A system can maintain several different audit trails concurrently. There are typically two kinds of audit records, the first is an event oriented log and the second is a record of every keystroke[83]. Event-based logs usually contain records describing either *system* events, *application* events, or *user* events. Each type of audit provides an important service, as described below.

18.3.1 Types of Audit Events

System audit records are generally used to monitor and fine tune system performance. *Application audit trails* may be used to discern flaws in applications, or violations of security policy committed within that application. *User audits* are generally used to hold individuals accountable for their actions. User audits may expose a variety of security violations. These might range from simple browsing to attempts to plant Trojan horses or gain unauthorized privileges.

The system itself enforces certain aspects of policy (particularly *system-specific* policy) such as access to files and access to the system itself. Monitoring the alteration of systems configuration files that implement the policy is important. If special accesses (e.g., security administrator access) must be used to alter configuration files, the system should generate audit records whenever these accesses are used.

[83] This type of log is referred to as *keystroke monitoring*.

A sample system log file showing authentication messages

```
Jan 27 17:14:04 host1 login: ROOT LOGIN console
Jan 27 17:15:04 host1 shutdown: reboot by root
Jan 27 17:18:38 host1 login: ROOT LOGIN console
Jan 27 17:19:37 host1 reboot: rebooted by root
Jan 28 09:46:53 host1 su: 'su root' succeeded for user1 on /dev/ttyp0
Jan 28 09:47:35 host1 shutdown: reboot by user1
Jan 28 09:53:24 host1 su: 'su root' succeeded for user1 on /dev/ttyp1
Feb 12 08:53:22 host1 su: 'su root' succeeded for user1 on /dev/ttyp1
Feb 17 08:57:50 host1 date: set by user1
Feb 17 13:22:52 host1 su: 'su root' succeeded for user1 on /dev/ttyp0
```

An application level audit record for a mail delivery system

```
Apr  9 11:20:22 host1 AA06370: from=<user2@host2>, size=3355, class=0
Apr  9 11:20:23 host1 AA06370: to=<user1@host1>, delay=00:00:02, stat=Sent
Apr  9 11:59:51 host1 AA06436: from=<user4@host3>, size=1424, class=0
Apr  9 11:59:52 host1 AA06436: to=<user1@host1>, delay=00:00:02, stat=Sent
Apr  9 12:43:52 host1 AA06441: from=<user2@host2>, size=2077, class=0
Apr  9 12:43:53 host1 AA06441: to=<user1@host1>, delay=00:00:01, stat=Sent
```

Sometimes when a finer granularity than system level auditing is required. *Application audit trails* can provide this greater level of recorded detail. If an application is critical, it can be desirable to record not only who invoked the application, but certain details specific to each invocation of the application. For example, consider an e-mail application. It may be desirable to record who sent mail, as well as to whom they sent mail and the length of messages. Another example would be that of a database application. It may be useful to audit who accessed what database as well as the individual rows or columns of a table that were read (or changed or deleted), instead of just recording the execution of the database program.

A *user audit trail* monitors and logs user activity in a system or application by recording events initiated by the user (e.g., access of a file, record or datafield, use of a modem, and so forth).

18.3.2 What Events Should be Recorded?

Whether a system-level audit trail monitors and logs events within applications (or an application-level audit trail is used to augment a system audit trail) a critical feature of auditing is the flexibility of audit trails. Ideally (from a security point of view), a system administrator would have the ability to monitor all system and user activity, but could choose to log only certain functions at the system level, and within certain applications. The decision of how much to log (and later review) should be a function of application/data sensitivity and

A user log showing a chronological list of commands executed by users

```
rcp          user1  ttyp0     0.02 secs Fri Apr  8 16:02
ls           user1  ttyp0     0.14 secs Fri Apr  8 16:01
clear        user1  ttyp0     0.05 secs Fri Apr  8 16:01
rpcinfo      user1  ttyp0     0.20 secs Fri Apr  8 16:01
nroff        user2  ttyp2     0.75 secs Fri Apr  8 16:00
sh           user2  ttyp2     0.02 secs Fri Apr  8 16:00
mv           user2  ttyp2     0.02 secs Fri Apr  8 16:00
sh           user2  ttyp2     0.03 secs Fri Apr  8 16:00
col          user2  ttyp2     0.09 secs Fri Apr  8 16:00
man          user2  ttyp2     0.14 secs Fri Apr  8 15:57
```

should be decided by each functional manager/application owner with guidance from the system administrator and the computer security manager/officer, weighing the costs and benefits of the logging.[84]

18.3.2.1 System-level Audit

If a system-level audit capability exists, the audit should capture, at a minimum, any attempt to logon (successful or unsuccessful), the logon ID, date and time of each logon attempt, date and time of each logoff, the devices used, and the function(s) performed once logged on (e.g., the applications that the user tried, successfully or unsuccessfully to invoke). System level logging also typically includes information that is not specifically security-related, such as system operations, cost accounting charges, and network performance.

> Some system-level audit trails cannot detect, and therefore, cannot log for later review, attempted logons. These audit trails can only monitor and log successful logons and subsequent activity. A system audit trail should be able to identify failed logon attempts, especially if the system does not limit the number of failed logon attempts. Without logging failed attempts, the only log entry is that of the intruder's successful logon, which would not appear unusual. To effectively detection intrusion, failed logon attempts are required.

18.3.2.2 Application-level Audit

System-level audit trails may not be able to track and log events *within* applications, or may not be able to provide the level of detail needed by application or data owners, the system administrator, or the computer security manager. Application-level audit trails should be able to monitor and log user activities including data files opened and closed, specific actions such as reading, editing and deleting records or fields, and printing reports. Some applications may be sensitive enough from a data availability, confidentiality, and/or integrity perspective

[84] Note that, in general, audit logging can have privacy implications. Users should be aware of applicable privacy laws, regulations, and policies that may apply in such situations.

that a "before" and "after" picture of each modified record (or the data element(s) changed within a record) should be captured by the audit trail.

18.3.2.3 User Audit

User audit trails can usually log:

- all commands directly initiated by the user;
- all identification and authentication attempts; and
- files and resources accessed.

It is most useful if options and parameters are also recorded from commands. It is much more useful to know that a user tried to delete a log file (e.g., to hide unauthorized actions) than to know the user merely issued the delete command, possibly for a personal data file.

18.3.3 What should be included in an audit trail record?

An audit trail must include sufficient information to establish what events occurred and who (or what) caused them. In general, an event record should specify when the event occurred, the user ID associated with the event, the program or command used to initiate the event, and the result. Date and time can help determine if the user was a masquerader or the actual person specified.

Audit Logs for Physical Access

Physical access control systems (e.g., a card/key entry system or an alarm system) use software and audit trails similar to general purpose computers. The following are *examples* of criteria that may be used in selecting which events to log:

- Date and time the access was attempted or made, the gate or door through which the access was attempted or made, and the individual (or user ID) making the attempt to access the gate or door.

- Invalid attempts should be monitored and logged by non-computer audit trails just as they are for computer systems audit trails. Management should be made aware if someone attempts to gain access during unauthorized hours.

- Logged information should also include attempts to add, modify or delete physical access privileges (e.g., granting a new employee access to the building or granting transferred employees access to their new office [and, of course, deleting their old access, as applicable]).

- As with system and application audit trails, auditing of non-computer functions can be implemented to send messages to security personnel indicating valid or invalid attempts to gain access to controlled spaces. In order not to desensitize a guard or monitor, all access should not result in messages being sent to a screen. Only exceptions, such as failed access attempts should be highlighted to those monitoring access.

18.4 Management Issues

The best planned and implemented audit trail is of no value without *timely review* of the logged data. Audit trail review may be performed periodically, as needed (often triggered by occurrence of a security event), automatically in real-time, or in some combination.

The following are examples of management issues that may have to be addressed when using audit trails.

18.4.1 Tools for Audit Trail Analysis

Many types of tools have been developed to help to reduce the amount of information contained in audit records, as well as to distill useful information from the raw data. Some of the types of tools are list below.

18.4.1.1 Audit Reduction Tools

Audit reduction tools are pre-processors designed to reduce the volume of audit records to facilitate manual review. Before a security review, many audit records known to have little security significance can be removed. (This decision alone may cut the number of records in the audit trail in half.) These tools generally remove records generated by specified classes of events. For example, records generated by nightly backups might be ignored for a policy enforcement audit.

18.4.1.2 Trends/Variance Detection Tools

It is possible to construct more sophisticated processors that monitor usage trends and detect major variations. For example, if a user typically logs in at 9 a.m., but appears at 4:30 a.m. one morning, this may indicate a security problem that may need to be investigated.

18.4.1.3 Attack Signature Detection Tools

An *attack signature* is a specific sequence of events indicative of an unauthorized access attempt. A simple example would be repeated failed login attempts. Tools can be used to detect such unauthorized activity.

18.4.2 When To Use the Audit Trails

Audit trails can be used to review what occurred after an event, for periodic reviews, and for real-time analysis.

18.4.2.1 Audit Trail Review After an Event

Following a known system or application software problem, a known violation of existing requirements by a user, or some unexplained system or user problem, the appropriate system-level or application-level administrator should review the audit trails. Review by the application/data owner would normally look at a separate report, based upon audit trail data, to determine if their resources are being misused.

Regardless of who is reviewing the report, to be able to do so effectively, it is important they know for what to look. Reviewers must understand what normal activity or transactions look like in order to be able to spot unusual activity. Audit trail review can be made easier if the audit trail function can be queried by user ID, terminal ID, application name, date and time, or some other set of parameters to run reports of selected information.

18.4.2.2 Periodic Review of Audit Trail Data .

Application owners, data owners, system administrators, data processing function managers, and computer security managers should determine how much review of audit trail records is necessary based on the importance of identifying unauthorized activities. This determination should have a direct correlation to the frequency of periodic reviews of audit trail data.

18.4.2.3 Real-Time Audit Analysis

Traditionally, audit is performed in a batch mode at regular intervals (e.g., daily). Audit records are archived during that interval for later analysis. Audit analysis tools can also be used in a real-time, or near real-time fashion. Such intrusion detection tools are based on audit reduction, attack signatures and variance techniques.

18.4.2.4 Audit Trail Monitoring

Manual review of audit records in real-time is almost never feasible on large multi-user systems due to the volume of records generated. However, it might be possible to view all records associated with a particular user or application, and view them in real time.[85]

18.4.3 Protecting Audit Trail Data

Audit trail data requires protection since it must be available for use when needed and is not useful if it is not accurate. Additionally, depending upon the specific implementation, there may be privacy concerns requiring protection of the confidentiality of audit trail data.

18.4.3.1 Maintaining Audit Trail Data Integrity, Availability, and Confidentiality

Access to on-line audit logs must be strictly controlled. Computer security managers and system administrators/managers should have access for review purposes. However, security and/or administration personnel who maintain logical access functions may have no need for access to audit logs.

It is particularly important to ensure the *integrity* of audit trail data against modification. One way to do this is to use digital signatures. (See the Cryptography Chapter.) Another way to do this is to use write-once devices. The audit trail files must be protected since, for example, intruders may try to "cover their tracks" by modifying audit trail records. Audit

---

[85] This is similar to keystroke monitoring, though, and may be legally restricted.

trail records should be protected by strong access controls to help prevent unauthorized access. The integrity of audit trail information may be particularly important when legal issues arise, such as preserving audit trail information for use as legal evidence. (This may, for example, requiring daily printing of the logs and signing them.) Questions of such legal issues should be directed to the cognizant legal authority.

Protection of the confidentiality of audit trail information may also be necessary. This may be required if the audit trail is recording information about users which may be sensitive or information regarding transaction data what may contain personal information (e.g., "before" and "after" records of modification to income tax data.) Strong access controls and encryption can be particularly effective in preserving confidentiality.

18.4.3.2 Retention of Audit Trail Data

System managers and administrators, with guidance from computer security personnel, should determine how long audit trail data will be maintained -- either on the system or in archive files.

18.5 Keystroke Monitoring[86]

> Keystroke monitoring is the process used to view or record both the keystrokes entered by a computer user and the computer's response during an interactive session.

Keystroke monitoring is usually considered a special case of auditing. Keystroke monitoring is the recording or viewing of users' keystrokes and the computer's responses. Examples of keystroke monitoring would include viewing characters as they are typed by users, reading users' electronic mail, and viewing other recorded information typed by users. Some forms of routine system maintenance may record user keystrokes. This could constitute keystroke monitoring if the keystrokes are preserved along with the user identification so that an administrator could determine the keystrokes entered by specific users. Keystroke monitoring is conducted in an effort to protect systems and data from intruders who access the systems without authority or in excess of their assigned authority. Monitoring keystrokes typed by

[86] The Department of Justice has advised that an ambiguity in U.S. law makes it unclear whether keystroke monitoring is considered equivalent to an unauthorized telephone wiretap. The ambiguity results from the fact that current laws were written years before such concerns as keystroke monitoring or system intruders became prevalent. Additionally, no legal precedent has been set to determine whether keystroke monitoring is legal or illegal. System administrators conducting such monitoring might be subject to criminal and civil liabilities. The Department of Justice advises system administrators to protect themselves by giving notice to system users if keystroke monitoring is being conducted. Notice should include agency/organization policy statements, training/awareness on the subject, and a banner notice on each system being monitored. [CSL Bulletin - March 1993]

intruders can help administrators assess and repair damage caused by intruders.

18.6 Interdependencies

Policy. The most fundamental interdependency of audit is with policy. Policy dictates who is authorized access to what system resources and therefore, specifies (directly or indirectly) what violations of policy should be identified through auditing.

Identification and Authentication. Audit trails are tools often used to help hold users accountable for their actions. To do this the users must be known to the system (usually accomplished through the identification and authentication process). However, as mentioned earlier, audit trails record events and associate them with the *perceived* user (i.e., the user ID). If a user is a impersonated, the audit trail will establish events but *not* the identity of the user.

Logical Access Control. Logical access controls are used to restrict the use of system resources to authorized users. Audit trails complement this in two ways. First, they are used to identify breakdowns in logical access controls. Audit trails may be used to verify that access control restrictions are behaving as expected. If a particular user is erroneously included in a group permitted access to a file, for example, audit trails can be used to identify this implementation error. Secondly, they are used *to audit use of resources by those who have legitimate access*.

Additionally, to protect audit trail files, access controls are used (e.g., to ensure that audit trails are not forged).

Contingency Planning. Audit trails assist in contingency planning by leaving a record of activities performed on the system or within a specific application. In the event of a technical malfunction occurs, this log can be used to help reconstruct the state of the system (or specific files).

Incident Response. If a security incident such as a hacker audit records and other intrusion detection methods can be used to help determine the extent of the incident. For example, was just one file browsed or was a trojan horse planted to collect passwords?

Cryptography. Digital signatures can be used to protect audit trails from undetected modification. (This does not prevent deletion or modification of the audit trail, but will provide an alert that the audit trail has been altered.) Digital signatures can also be used in conjunction with adding secure time stamps to audit records. Encryption can be used if confidentiality of audit trail information is important.

18.7 Cost Considerations

Auditing is not inexpensive. Some system overhead is incurred recording the audit trail. The actual system overhead will depend upon the level of auditing used; storing and processing more records (and more detailed records) makes auditing more expensive. The second cost involves human and machine time required to do the analysis. This can be minimized by using tools to perform most of the analysis. Many simple analyzers can be constructed quickly (and cheaply) from system utilities. These simple analyzers are limited to audit reduction and identifying particularly sensitive events. More complex tools which identify trends or sequences of events are slowly becoming available as off-the-shelf software. (If complex tools are not available for a system, development may be prohibitively expensive. Some intrusion detection systems, for example, have taken years to develop.)

The final cost of auditing is the cost of investigating anomalous events. If the system is identifying too many events as suspicious, administrators may spend undue time reconstructing events and questioning personnel.

18.8 Summary

Auditing can assist in securing computer resources by helping to provide and maintain individual accountability, reconstruction of events, intrusion detection, and analysis of problems. Auditing can be accomplished at the system, application and individual user level. In deciding how to implement auditing, management must address a number of issues, including how and when audit trails will be used and how they will be protected.

18.9 References

A Guide to Understanding Audit in Trusted Systems, NCSC-TG-001, Version-2. Ft. Meade, MD: National Computer Seurity Center, 1988.

Enger, Norman L. and Paul W. Howerton. "Computer Security." AMACOM, 1980.

Guideance on the Legality of Keystroke Monitoring. CSL Bulletin. National Institute of Standards and Technology, March 1993.

Kim, G. and E. Spafford, "Monitoring File System Integrity on UNIX Platforms." *Infosecurity News*. 4(4), 1993, pp. 21-22.

Lunt, T. "Automated Audit Trail Analysis for Intrusion Detection," *Computer Audit Update*, April, 1992, pp. 2-8.

Phillips, P. W. "New Approach Identifies Malicious System Activity." *Signal.* 46(7), 1992. pp. 65-66.

Ruthberg, Z. and et.al., *Guide To Auditing for Controls and Security: A System Development*

Life Cycle Approach. Spec. Pub. 500-153. National Institute of Standards and Technology, 1988.

Stoll, Clifford. *The Cuckoo's Egg*. Doubleday, 1989.

19 Cryptography

19.1 Introduction

Cryptography provides an important tool for the protection of information and is used in many aspects of computer security. For example, cryptography can help provide data confidentiality, integrity, electronic signatures, and advanced user authentication. Many people realize that modern cryptography relies upon advanced mathematics. Not all understand, however, that users can obtain the benefits offered by cryptography without an understanding of its mathematical underpinnings.

Cryptography's uses are wide and varied. This chapter describes cryptography as a tool for satisfying a wide spectrum of computer security needs and requirements. It describes fundamental aspects of the basic cryptographic technologies and some specific ways cryptography can be applied to improve security. The chapter also explores some of the important issues which should be considered when incorporating cryptography into computer systems.

> Cryptography is traditionally associated only with keeping data secret. However, modern cryptography can be used to provide many security services such as electronic signatures and ensuring that data has not been modified.

19.2 Basic Cryptographic Technologies

Cryptography relies upon two basic components: an *algorithm* (or cryptographic methodology) and a *key*. In modern cryptographic systems, the algorithms are complex mathematical formulae and keys are strings of bits. For two parties to communicate, they must use the same algorithm (or algorithms which are designed to work together). In some cases, they must also use the same key. Many cryptographic keys must be kept secret; sometimes algorithms are also kept secret.

> There are two basic types of cryptography: "secret key" and "public key."

There are two basic types of cryptography: *secret key systems* (also called symmetric systems) and *public key systems* (also called asymmetric systems). Table 1 compares some of the distinct features of secret and public key systems. Both types of systems offer advantages and disadvantages. Often, the two are combined to form a *hybrid system* to exploit the strengths of each type. In order to determine which type of cryptography to use, an organization must first identify its security requirements and operating environment. Then the organization can determine which type of cryptography best meets its needs.

	SECRET KEY CRYPTOGRAPHY	PUBLIC KEY CRYPTOGRAPHY
NUMBER OF KEYS	Single key.	Pair of keys.
TYPES OF KEYS	Key is secret.	One key is private and one key is public.
PROTECTION OF KEYS	Disclosure and modification.	Disclosure and modification for private keys and modification for public keys.
RELATIVE SPEEDS	Faster.	Slower.

19.2.1 Secret Key Cryptography

> Secret key cryptography has been in use for centuries. Early forms of secret key cryptography merely transposed the written characters in order to hide the message.

In secret key cryptography, two (or more) parties share the same key and that key is used to encrypt and decrypt data. As the name implies, secret key cryptography relies on keeping the key secret. If the key is compromised, the security offered by cryptography is severely reduced or eliminated. Secret key cryptography assumes that the parties who share a key rely upon each other not to disclose the key and protect it against modification.

The best known secret key system is the *Data Encryption Standard* (DES), published by NIST as Federal Information Processing Standard (FIPS) 46-2. Although the adequacy of DES has at times been questioned, these claims remain unsubstantiated, and DES remains strong. It is the most widely accepted, publicly available cryptographic system today. DES is a federal standard which has been widely adopted by the private sector. The American National Standards Institute (ANSI) has adopted DES as the basis for encryption, integrity, access control, and key management standards.

The *Escrowed Encryption Standard*, published as FIPS 185, also makes use of a secret key system. (See discussion of key escrow encryption in the Encryption section of this chapter.)

19.2.2 Public Key Cryptography

Public key cryptography is a modern invention and requires the use of advanced mathematics.

Whereas secret key cryptography employs a single key shared by two (or more) parties, public key cryptography uses a pair of keys for *each* party. One of the keys of the pair is "public" and one "private." The public key can be made known to other parties. The private key must be kept confidential and be known only to its owner. Both keys, however, must be protected against modification.

Public key cryptography is particularly useful in situations when the parties wishing to communicate cannot rely upon each other or do not share a common key. There are several public key cryptographic systems. One of the first public key systems, named RSA after its three MIT creators, Ronald Rivest, Adi Shamir, and Len Adleman, is in wide use and can provide many different security services. The Digital Signature Standard (DSS), which is described later in the chapter, is another example of a public key system.

19.2.3 Hybrid Cryptographic Systems

Public and secret key cryptography have relative advantages and disadvantages. Public key cryptography does not require users to share a common key; however secret key cryptography is much faster. Equivalent implementations of secret key cryptography can run 1,000 to 10,000 times faster than public key cryptography.

Secret key systems are often used for bulk data encryption and public key systems are often used for automated key distribution.

To maximize the advantages of both secret and public key cryptography, a computer system can use both types in a complementary manner, with each performing different functions. Typically, the speed advantage of secret key cryptography means that it is used for encrypting data. Public key cryptography is used for applications that are less demanding to a computer system's resources such as encrypting the keys used by secret key cryptography (for distribution) or to sign messages. These types of hybrid systems provide many of the advantages of both public and secret key cryptography while minimizing the disadvantages.

19.3 Uses of Cryptography

As discussed in the Introduction, cryptography can be used to provide data confidentiality and integrity. It can also be used to create electronic signatures and as a basis for identification and authentication.

Cryptography is used to protect data *both* inside and outside of the boundaries of a computer system. Outside the computer system, cryptography is sometimes the *only* way to protect data. While data is in a computer system, it is normally protected with logical and physical access controls (perhaps supplemented by cryptography). However, when data is in transit across communications lines or resident on someone else's computer, it cannot be protected by the originator's[87] logical or physical access controls. Cryptography provides a solution by protecting data even when it is no longer in the control of the originator.

19.3.1 Data Encryption

One of the best ways to obtain cost-effective data confidentiality is through the use of encryption. Encryption transforms intelligible data, called *plaintext*[88], into an unintelligible form, called *ciphertext*. This process is reversed through the process of decryption. Once data is encrypted, the ciphertext does not have to be protected against disclosure. However, if ciphertext is modified, it will not decrypt correctly.

```
                 Original Message:
          "Send the money on Friday."
                        |
              +-----------------+
      Key->   |   Encryption    |
              +-----------------+
                        |
               Encrypted Message:
          "Drmf\yjr\,pmru\pm\Gtofsu/"
                        |
              +-----------------+
      Key->   |   Decryption    |
              +-----------------+
                        |
        Original (decrypted) message.
          "Send the money on Friday."

                   Secret Key
              Encryption/Decryption
```

Both secret key and public key cryptography can be used for data encryption although not all public key algorithms provide for data encryption.

In order to use a secret key algorithm, data is encrypted using a key. The same key must be used to decrypt the data. In secret key systems, the key may be used only once or may be used many times.

When public key is used for encryption, any party can use any other party's public key to encrypt message; however, only the party with the corresponding private key can decrypt, and thus read, the message. For example, public key cryptography can be used to send an encrypted confidential message between Person A and Person B. Each person has two keys, one public and one private. Person A can encrypt a message to Person B by using Person B's public key. Only Person B can decrypt the message, which requires use of Person B's private key. This ensures that only Person B can read the message, thus providing data confidentiality.

Since secret key encryption is typically much faster, it is normally used for encrypting larger

---

[87] The originator does not have to be the original creator of the data. It can also be a guardian or custodian of the data.

[88] Plaintext can be intelligible to a human (e.g., a novel) or to a machine (e.g., executable code).

amounts of data.

```
                     Message Prepared by
                          Person A
                             |
                      +------------+
Person B's ----->     | Encryption |
public key            +------------+
                             |
                     Encrypted Message
                 Transmitted to Person B
                             |
                      +------------+
Person B's ----->     | Decryption |
private key           +------------+
                             |
                     Plaintext Message
                     Read by Person B

Person A knows that only Person B can
read the message.

         Use of Public Key
     for Encryption/Decryption
```

Key Escrow. Because cryptography can provide extremely strong encryption, it can thwart the government's efforts to lawfully perform electronic surveillance. For example, if strong cryptography is used to encrypt a phone conversation, a court-authorized wiretap will not be effective. In order to meet the needs of the government *and* provide privacy, the Federal Government has adopted key escrow cryptography. This technology allows the use of strong encryption, but also allows the government to obtain keys, when legally allowed, that can be used to decrypt data. These decryption keys are held by escrow agents who only release the keys when it is legally authorized. NIST has published the *Escrowed Encryption Standard* as FIPS PUB 185. Under the Federal Government's key escrow initiative, the decryption keys are split into parts and given to separate escrow authorities. Access to one part of the key does *not* help decrypt the data; both keys must be obtained.

19.3.2 Integrity

In computer systems, it is not always possible for humans to scan information to determine if data has been erased, added or modified. Even if scanning were possible, the individual may have no way of knowing what the correct data should be. For example, "do" may be changed to "do not," or $1,000 may be changed to $10,000. It is therefore desirable to have an automated means of detecting *both* intentional and unintentional modifications of data. While error detecting codes have long been used in communications protocols (e.g., parity bits), these are easily defeated by adversaries to allow modifications to go undetected. Cryptography can effectively provide detection of both intentional and unintentional modification. It is important to point out that cryptography does not protect files from being modified, but it does alert the user that the data has been modified. Both secret key and public key cryptography can be used to provide integrity, although newer public key methods may offer more flexibility than the older secret key method. Secret key integrity verification systems, however, have been successfully integrated into many applications.

When secret key cryptography is used, a message authentication code (MAC) is calculated from and appended to the data. To verify that the data has not been modified at a later time, any party with access to the correct secret key can recalculate the MAC. The new MAC is compared with the original MAC, and if they are identical, the verifier has confidence that the data has not been modified by an unauthorized party. FIPS 113, *Computer Data Authentication*, specifies a standard technique for calculating a Message Authentication Code (MAC) for integrity verification.

Public key cryptography provides integrity verification through the use of public key signatures and secure hashes. A secure hash algorithm is used to create a message digest. The message digest, called a hash, is a short form of the message that changes if the message is modified. The hash is then signed with a private key. Anyone can recalculate the hash and use the corresponding public key to verify the integrity of the message.[89]

19.3.3 Electronic Signatures

Today's computer systems store and process increasing numbers of paper-based documents in electronic form. Having documents in electronic form permits rapid processing and transmission and improves overall efficiency. However, approval of a written document has traditionally been indicated by a written signature. What is needed, therefore, is the electronic equivalent of a written signature which can be recognized as having the same legal status as a written signature. In addition to the integrity protections, discussed above, cryptography can provide a means of linking a document with a particular person, as is done with a written signature.

What is an electronic signature?

An electronic signature performs a similar function to a written signature. It is used to verify the origin and contents of a message. For example, a recipient of data (e.g., an e-mail message) can verify who signed the data and that the data was not modified after it was signed. This also means that the originator (e.g., sender of an e-mail message) cannot falsely deny having signed the data.

Why not just take a digital picture of a written signature? Unfortunately, a digital image of a written signature (also known as a digiti*zed* written signature) does not provide adequate security. Such a digitized written signature could easily be copied from one electronic document to another with no way to determine whether or not it is legitimate. Use of cryptography, however, provides a solution.

While they have many uses, electronic signatures have particularly important implications for business applications, which have traditionally relied on written signatures. Other applications that use written authorizations as well as applications that previously relied on less secure authorizations will also benefit.

Electronic signatures can use either secret key or public key cryptography. Public key

[89] Sometimes a secure hash is used for integrity verification. However, this can be defeated if the hash is not stored in a secure location, since it may be possible for someone to change the message and then replace the old hash with a new one based on the modified message.

methods are generally easier to use for electronic signature. Secret key systems were developed earlier and require additional controls but do provide strong electronic signatures.

Both types of signature rely on the secrecy of keys, either the secret key or the private key. If a key is compromised (by theft, coercion, or trickery), then the electronic originator of a message may not be the same as the owner of the key.[90]

19.3.3.1 Secret Key

> Systems incorporating message authentication technology have been approved for use by the federal government as a replacement for written signatures on electronic documents.

An electronic signature can be implemented using secret key message authentication codes (MACs). For example, if two parties share a secret key and one party receives data with a MAC that is correctly verified using the shared key, that party may assume that the data was sent by the other party. This assumes, however, that the two parties trust each other. Thus, through the use of a MAC, in addition to data integrity, a form of electronic signature is obtained. Using additional controls, such as key notarization and key attributes, it is possible to provide an electronic signature even if the two parties do not trust each other.

19.3.3.2 Public Key

Another type of electronic signature called a *digital signature* is implemented using public key cryptography. Data is electronically signed by applying the originator's private key to the data. (The exact mathematical process for doing this is not important for this discussion.) To increase the speed of the process, the private key is applied to a shorter form of the data, called a "hash" or "message digest," rather than to the entire set of data. The resulting digital signature can be stored or transmitted along with the data. The signature can be verified by any party using the public key of the signer. This feature is very useful, for example, when distributing signed copies of virus-free software. Any recipient can verify that the program remains virus-free. If the signature verifies properly, then the verifier has confidence that the data was not modified after it was signed and that it was signed by the owner of the public key.
NIST has published standards for a digital signature and a secure hash for use by the federal government in FIPS 186, *Digital Signature Standard* and FIPS 180, *Secure Hash Standard*.

19.3.4 User Authentication

Cryptography can increase security in user authentication techniques. As discussed in the

[90] Although the binding of cryptographic keys to actual people is a significant problem, it does not necessarily make electronic signatures less secure than written signatures. Trickery and coercion are problems for written signatures as well. In addition, written signatures can be forged.

Identification and Authentication chapter, cryptography is the basis for several advanced authentication methods. Instead of communicating passwords over an open network, authentication can be performed by demonstrating knowledge of a cryptographic key. Using these methods, a onetime password, which is not susceptible to eavesdropping, can be achieved. User authentication can use either secret or public key cryptography.

```
                         Message Prepared by
                             Person A
                                 |
                          +-------------+
Person A's ----->         |             |
private key               |  Signature  |
                          |             |
                          |             |
                          +-------------+
                                 |
                    Transmitted to Person B
                                 |
                          +-------------+
                          |             |
                          |Verification |
                          |             |
Person A's ----->         |             |
public key                +-------------+
                                 |
                         Message Verified
                         Read by Person B

Person B knows that only Person A could
have sent the message.
```

Use of Public Key Cryptography for Digital Signature

19.4 Considerations in Utilizing Cryptography

This section explores several important issues which should be considered when utilizing (e.g., designing, implementing, integrating, using) cryptography in a computer system.

19.4.1 Standards

NIST and other standards organizations have developed numerous standards for the design, implementation, and use of cryptography and for its integration into automated systems. The use of cryptographic standards can provide significant benefits to an organization. Standards provide solutions that have been accepted by a wide community and that have been reviewed by experts in relevant areas. Standards help ensure interoperability among different vendors' equipment, thus allowing an organization to select from among various products in order to find cost-effective equipment. By using standards, organizations can reduce costs and protect their investments in technology.

Managers and users of computer systems must select among various standards when deciding to utilize cryptography. Selection should be based on cost/effectiveness analysis, trends in standard's acceptance and interoperability requirements. In addition, the applicability section of each standard should be carefully analyzed to determine if that standard is applicable to the organization and the desired application. For example, the Data Encryption Standard and the Escrowed Encryption Standard are both applicable to certain applications involving communications of data over commercial modems. Some federal standards are mandatory for federal computer systems.

> Applicable security standards provide a common level of security and interoperability among users.

19.4.2 Hardware vs. Software Implementations

Cryptography can be implemented in either hardware or software. Each has its related costs and benefits. In general software is less expensive and slower. Although for large applications, hardware may be less expensive. In addition, software may be less secure since it is more easily modified or bypassed than equivalent hardware products. Tamper resistance is usually considered better in hardware. The trade-offs among security, cost, simplicity, efficiency and ease of implementation must be studied by managers acquiring various security products meeting a standard.

In many cases, cryptography is implemented in a hardware device (e.g., electronic chip, ROM protected processor) but is controlled by software. This software requires integrity protection to assure that the hardware device is provided with correct information (i.e., controls, data) and is not bypassed. Thus a hybrid solution is generally provided even when the basic cryptography is implemented in hardware. Effective security requires the correct management of the entire hybrid solution.

19.4.3 Key Management

The proper management of cryptographic keys is essential to the effective use of cryptography for security. Ultimately, the security of information protected by cryptography is directly dependent upon the protection afforded to the keys. Key management involves the procedures and protocols, both manual and automated, to be used throughout the entire life cycle of the keys, which includes the generation, distribution, storage, entry, use, destruction and archiving of the cryptographic keys. Protection must be provided to safeguard all keys from modification. In addition, secret keys and private keys must be protected from unauthorized disclosure.

With secret key cryptography, the secret key(s) must be securely distributed (i.e., protected from replacement, modification and disclosure) to the parties wishing to communicate. Depending upon the number and location of users, this may not be a trivial task. Automated techniques for generating and distributing cryptographic keys can ease the overhead of key management, but some resources must be devoted to this task. FIPS 171, *Key Management Using ANSI X9.17*, provides key management solutions for a variety of operational environments.

Public key cryptography users also must satisfy certain key management requirements. For example, since a private-public key pair is associated (i.e., generated or held) with a specific user, it is necessary to "bind" the public part of the key pair to the user.[91] This is done by a "certification authority," which verifies the identity of an individual before "certifying" the correct "binding" of the user with the public key. In addition, within any community utilizing

---

[91] In some cases, the key may be bound to a position or an organization rather than to an individual user.

public key cryptography, the secure creation, binding, and verification of public keys as well as protection of all keys is critical. In order to create, bind, and verify keys for a wide variety of users, a public key infrastructure is needed.

19.4.4 Security of Cryptographic Modules

Cryptography is typically implemented in a *module* comprised of software, firmware, hardware, or some combination thereof. This module contains the cryptographic algorithm(s), certain control parameters and temporary storage facilities for the key(s) being used by the algorithm(s). To provide adequate security, the module must be protected from tampering. Protection may also have to be provided to protect the algorithm against disclosure. Users must be able to rely on the proper functioning of the cryptography which requires the secure design, implementation and use of a cryptographic module.

FIPS 140-1, *Security Requirements for Cryptographic Modules*, specifies the physical and logical security requirements for a cryptographic module. The standard defines four security levels for cryptographic modules, with each level providing a significant increase in security over the preceding level. The four levels of security allow for cost-effective solutions that are appropriate for different degrees of data sensitivity and different applications environments. The user can select the best module for any given application or system, avoiding the cost of unnecessary security features.

19.4.5 Networking Issues

The use of cryptography within networking applications may require special considerations. In these applications, the suitability of a cryptographic module may depend on its capability of handling special requirements imposed by locally attached communications equipment or by the network protocols and software.

Encrypted information, MACs, or digital signatures may require transparent communications protocols or equipment to avoid being misinterpreted by the communications equipment or software as being control information. It may be necessary to format the encrypted information, MAC, or digital signature to ensure that it does not confuse the communications equipment or software. It is essential to ensure that cryptography satisfies the requirements imposed by the communications equipment and does not interfere with the proper and efficient operation of the network.

19.4.6 Export

The United States Government controls the export of cryptographic implementations. The rules governing export can be quite complex since they consider multiple factors. In addition, cryptography is a rapidly changing field, and rules may change from time to time. Questions concerning the export of a particular implementation should be addressed to appropriate legal counsel.

19.5 Cost Considerations

The cost of employing cryptography to protect information can be characterized in terms of both direct and indirect costs. Cost is determined in part by product availability; a wide variety of products exist for implementing cryptography in integrated circuit (IC) chips, add-on boards or adapters, and stand-alone units.

19.5.1 Direct Costs

The direct costs of employing cryptography include:

- acquiring or implementing the cryptographic module and integrating it into the computer system; the medium (i.e., hardware, software, firmware or combination) and various other issues such as level of security, logical and physical configuration, and special processing requirements will have an impact on cost;

- managing the cryptography, and, in particular, managing the cryptographic keys, which includes key generation, distribution, archiving and disposition as well as security measures to protect the keys, as appropriate.

19.5.2 Indirect Costs

The indirect costs of employing cryptography include:

- a small decrease in system or network performance, resulting from the additional overhead of applying cryptographic protection to stored or communicated data;

- changes in the way users interact with the system, resulting from more stringent security enforcement. However, cryptography can be made nearly transparent to the users so that the impact is minimal.

19.6 Interdependencies

There are many interdependencies between cryptography and other security controls highlighted in this handbook. Cryptography both depends on other safeguards and assists in providing other security safeguards.

19.6.1 Physical Security

Physical protection of a cryptographic module is required to prevent, or at least detect, physical replacement or modification of the cryptographic system and keys within it. In many environments (e.g., open offices, portable computers), the cryptographic module itself must provide the desired levels of physical security. In other environments (e.g., closed communications facilities, steel encased Cash Issuing Terminals), a cryptographic module may be employed within a secured facility with adequate physical security for the cryptographic module.

19.6.2 User Authentication

Cryptography can be used both to protect passwords that are stored in computer systems and to protect passwords that are communicated between computers. Furthermore, cryptographic-based authentication techniques may be used in conjunction with, or in place of, password-based techniques to provide stronger authentication of users.

19.6.3 Logical Access Control

In many cases, cryptographic software may be embedded within a host system and it may not be feasible to provide extensive physical protection to the host system. In these cases, logical access control may provide a means to isolate the cryptographic software from other parts of the host system and for protecting the cryptographic software from tampering and the keys from replacement or disclosure. The use of such controls should provide the equivalent of physical protection.

19.6.4 Audit

Cryptography may play a useful role in auditing. For example, audit records may need to be signed. Cryptography may also be needed to protect audit records stored on computer systems from disclosure or modification.

Conversely, a cryptographic system should be monitored and periodically audited to assure that it is satisfying its security objectives. All parameters associated with correct operation of the cryptographic system should be reviewed. Operation of the system itself should be periodically tested and the results audited. Certain information, such as private and secret keys, should not be subject to audit. However, non-secret or non-private keys could be used in a simulated audit procedure.

19.6.5 Assurance

Assurance that a cryptographic module is properly and securely implemented is essential to the effective use of cryptography. NIST maintains validation programs for several of its standards for cryptography. Vendors can have their products validated for conformance to the standard through a rigorous set of tests. Such testing provides increased assurance that a module meets stated standards, and system designers, integrators, and users can have greater confidence that validated products conform to accepted standards.

> NIST maintains validation programs for several of its cryptographic standards.

In addition to standard validation programs, other forms of assurance may be desirable. Security evaluations are generally more encompassing than standard's validations, including analysis of non-standard parts of the system, interactions among various parts of the system and fault handling within the system. Monitoring programs provide additional assurance that the security system is functioning properly.

19.7 Summary

Cryptography is the only known mechanism (i.e., technical safeguard) that can provide certain types of security in computer systems, namely protecting information from unauthorized disclosure in a distributed computer system or network. It can be a highly cost-effective way to protect data from intentional modification especially in networks. User authentication procedures can be greatly strengthened through cryptographic techniques. Use of digital signatures provides both user authentication and message integrity assurance. While there are direct costs (cryptographic modules, software modifications) and indirect costs (communications overhead) associated with the use of cryptography, integration of standard cryptographic algorithms and automated key management systems into new computer products and systems will minimize these costs and greatly improve the security of the resulting systems.

19.8 References

"Advanced Authentication Technology." Computer Systems Laboratory Bulletin. National Institute of Standards and Technology, November 1991.

Alexander, M. (editor). "Protecting Data with Secret Codes," *Infosecurity News*. 4(6), 1993. pp. 72-78.

American National Standard for Financial Institution Key Management (Wholesale). ANSI X9.17-1985. American Bankers Association, Washington, DC.

Computer Data Authentication. FIPS PUB 113. National Institute of Standards and Technology. May 30, 1985.

Data Encryption Standard. FIPS PUB 46-2. National Institute of Standards and Technology. December 30, 1993.

Denning, D. "Cryptography Overview and the Key-Escrow Chip." 1993.

Denning, P. and D. Denning, "The Clipper and Capstone Encryption Systems." *American Scientist.* 81(4), 1993, pp. 319-323.

Diffie, W. and M. Hellman. "New Directions in Cryptography." *IEEE Transactions on*

Information Theory. Vol. IT-22, No. 6, November 1976, pp. 644-654.

"Digital Signature Standard." Computer Systems Laboratory Bulletin. National Institute of Standards and Technology. January 1993.

Digital Signature Standard. FIPS PUB 186. National Institute of Standards and Technology. May, 1994.

Duncan, R. "Encryption ABCs." *Infosecurity News.* 5(2), 1994. pp. 36-41.

Escrowed Encryption Standard. FIPS PUB 185. National Institute of Standards and Technology. 1994.

Information Processing Systems - Open Systems Interconnection Reference Model - Part 2: Security Architecture. ISO 7498/2. International Organization for Standardization, 1988.

Key Management Using ANSI X9.17. FIPS PUB 171. National Institute of Standards and Technology. April 27, 1992.

Meyer, C.H. and S. M. Matyas, *Cryptography: A New Dimension in Computer Data Security.* New York, NY: John Wiley and Sons, 1982.

Nechvatal, James. *Public-Key Cryptography*. Spec. Pub. 800-2. National Institute of Standards and Technology, April 1991.

Rivest, R., A. Shamir, and L. Adleman. "A Method For Obtaining Digital Signatures and Public-Key Cryptosystems." *Communications of the ACM*, Vol. 21, No. 2, 1978, pp. 120-126.

Schneier, B. "A Taxonomy of Encryption Algorithms." *Computer Security Journal*. 9(1), 1193. pp. 39-60.

Schneier, B. "Four Crypto Standards." *Infosecurity News*. 4(2), 1993. pp. 38-39.

Secure Hash Standard. FIPS PUB 180. National Institute of Standards and Technology. May 11, 1993.

"Security Safeguards and Practices." *Defending Secrets, Sharing Data: New Locks and Keys for Electronic Information*, Office of Technology Assessment. U.S. Congress. Washington, D.C.: 1987, p. 54-72.

Security Requirements for Cryptographic Modules. FIPS PUB 140-1. National Institute of Standards and Technology. January 11, 1994.

Voydock, V.L. and S.T. Kent. "Security Mechanisms in High-Level Network Protocols." *ACM Computing Surveys*, Vol. 15, No. 2, June 1983.