

MANAGEMENT 414
SANS +S™
TRAINING PROGRAM
FOR THE CISSP®
CERTIFICATION EXAM

414.1

Overview and Access Control Systems and Methodology

Copyright © 2006, The SANS Institute. All rights reserved. The entire contents of this publication are the property of the SANS Institute. User may not copy, reproduce, distribute, display, modify or create derivative works based upon all or any portion of this publication in any medium whether printed, electronic or otherwise, without the express written consent of the SANS Institute. Without limiting the foregoing, user may not reproduce, distribute, re-publish, display, modify, or create derivative works based upon all or any portion of this publication for purposes of teaching any computer or electronic security courses to any third party without the express written consent of the SANS Institute.

Preface

The cardinal rule for SANS training is that after you take a course you should be able to apply what you learned directly the day you get back into the workplace. My journey into writing about the 10 Domains started when Stephen Northcutt asked that I lead the development of adding the ISC2 10 Domains of Knowledge into SANS Security Essentials. That is SANS most popular training course and when taught bootcamp style it does an amazing job of helping students become capable of using hands-on techie tools. However, there had been a split in the community whether Security Essentials, which favored technical and pragmatic material, or the ISC2 10 Domains, which favors theory, should be the baseline standard for an information security professional. We were discussing this hotly debated issue in the SANS faculty speaker room over lunch one day and it suddenly dawned on us, why not add the 10 Domains into Security Essentials? Tony Cole, CISSP, was assigned to evaluate Security Essentials and determined that about 60% of the 10 Domains material was already covered in Security Essentials. Clement Dupuis and I were the leads on the project. This was a very successful edit and a number of students have passed their CISSP exams after going through SANS Security Essentials with the ISC2 10 Domains. However, when we added the additional material there was no longer time to cover the application of the material to the workplace; in addition, there are some students who prefer the more formal 10 domain structure.

To best meet the needs of the students, SANS authorized the creation of Management 414, SANS CISSP® 10 Domains +S™, which covers the 10 Domains of Knowledge in a formal 10 domain structure. In the meantime, Clement Dupuis, Stephen Northcutt, Marcus Sachs, Bill Stearns and Joshua Wright are removing some of the 10 Domains material from SANS Security Essentials and returning it back to the original vision for that track, to fully cover the essentials of technical security. Moreover, SANS has insisted that the course teach the application of the 10 Domains in the workplace — something no other 10 Domains course, including ISC2's does. This course meets the SANS promise: what you learn in the course you will be able to apply in the work place. One of the most important things I have learned from Alan Paller, Director of Research, in the years I have been involved with SANS is the importance of community consensus. In order to provide the highest quality training we have recruited experts to review the material and come to consensus on the course content and the application of the information. With the help of Zoe Dias, SANS Faculty Director, we enlisted a total of 68 reviewers from 10 countries. All but two of the reviewers are active CISSP's. The main author for the course, Eric Cole, has been a CISSP for almost 10 years.

SANS enthusiastically applauds the expert work of our technical reviewers/editors:

Monica Anklam, CISSP No. 31995, USA
Alex Arndt, CISSP No. 52343, Canada
Hank Askin, CISSP No. 40792, USA
Anjali Atanacio, CISSP No. 27039, USA
Jason Bevis, CISSP No. 35285, USA
Ron Black, CISSP No. 24245, USA
Anton Bojanec, CISSP No. 24560, Slovenia
Olufremi Bolanle, CISSP No. 51582, Nigeria
Jeff Bontsas, CISSP No. 39135, USA
Derek Browne, CISSP No. 26099, Canada
Sherry Callahan, CISSP No. 21760, USA
Ed Capizzi, CISSP No. 35909, USA
Jim Cate, CISSP No. 37031, USA
Patrick Chan, CISSP No. 40222, Canada
Jerry Chen, CISSP No. 47413, Canada
Daniel Cline, CISSP No. 31366, USA
Chris Cook, CISSP No. 38254, UK

Edwin Covert, CISSP No. 3597, USA
Phil Curran, CISSP No. 31708, USA
Edgar Danielyan, CISSP No. 42834, UK/Armenia
David Dann, CISSP No. 51571, USA
Gary Delaney, CISSP No. 37636, USA
Sandeep Dhameja, CISSP No. 33585, USA
Joe Dial, CISSP No. 25358, USA
Heinz Durr, CISSP No. 42160, Switzerland
Darin Dutcher, CISSP No. 41299, USA
Rene Evers, CISSP No. 29057, USA
Chris Farrow, CISSP No. 45570, USA
Kenneth Fox, CISSP No. 42293, USA
Roger Fradenburgh, CISSP No. 28099, USA
Brian Freedman, CISSP No. 49504, USA
Donald Glass, CISSP No. 42244, USA
Mark Heinrich, CISSP No. 36190, USA

Lorna Hutcheson, USA
Lawrence Johnson, CISSP No. 25456, USA
Chaiw Kok Kee, CISSP No. 31589, Malaysia
Darrin Lau, CISSP No. 29948, USA
Eliot Leibowitz, CISSP No. 43782, USA
Steven Leong, CISSP No. 30313, Singapore
Chip Meadows, CISSP No. 10070, USA
Sean Mitchell, CISSP No. 36817, USA
Michael Morrell, CISSP No. 36227, USA
Pamela Nottage, CISSP No. 3758, USA
Sanjay Pandit, CISSP No. 44786, USA
John Pao, CISSP No. 29876, USA
Ariya Parsamanesh, CISSP No. 36074, AUS
Stephen Patton, CISSP No. 49746, USA
Robert Pfau, CISSP No. 21572, USA
Gabriel Proulx, CISSP No. 34018, Canada

Jim Purcell, CISSP No. 34519, USA
Andrew Salzman, CISSP No. 25162, USA
Amarottam Shrestha, CISSP No. 41671, AUS
Michael Solomon, CISSP No. 26517, USA
Robert Sorensen, CISSP No. 48304, USA
George Starcher, CISSP No. 34689, USA
Bruce Swartz, CISSP No. 46522, USA
David Taylor, CISSP No. 55890, USA
Brad Towers, CISSP No. 27957, USA
Jill Treu, CISSP No. 43196, USA
Tim Weil, CISSP No. 44250, USA
Deborah Weinstein, CISSP No. 44411, USA
Melody Wilson, CISSP No. 4130, USA
Steven Winterfield, CISSP No. 38096, USA
Kelli Wolfe, USA
Wayde York, CISSP No. 30404, USA

I have had the privilege of the best seat in the house and have really enjoyed working with the CISSP team. I sincerely hope that you benefit greatly from the information in these books and am very interested in your feedback. Please feel free to send me suggestions, corrections or questions to mqt414@sans.org.

Eric Cole, Senior Instructor and Research Fellow
The SANS Institute

SANS CISSP® 10 Domains

The SANS Institute

Welcome to the CISSP® 10 Domains class

CISSP Applicants

Applicants must have one of the following:

- Minimum of four years of direct experience
- Minimum of three years of direct experience with a four-year college degree

*A Master's degree from a Center of Excellence in security can substitute for one year of experience.

In order to become a CISSP applicants must have one of the following:

- Minimum of 4 years of direct experience
- Minimum of 3 years of direct experience with a 4-year college degree

It should be noted that a Master's degree from a Center of Excellence in security can substituted for one year of experience.

CISSP Overview

Two-Step Process

1. Pass exam with a score of 700 or higher
 - Exam consists of 250 multiple-choice questions
 - Each question has 4 choices .
 - 25 research questions that do not count
 - 6 hours to take the exam
 - Based on 10 domains of knowledge
 - Endorsement
 - Current CISSP must sign a form
2. (Optional) Audit
 - Submit resume for review

The CISSP course and exam are offered through ISC² and their Web site, which can be found at www.isc2.org. The exam is a multiple-choice exam that is offered at set times and locations throughout the year. The exam is not offered through a central testing center, such as an MCSE or Cisco exam.

To become a CISSP, you must meet these requirements:

- Pass the exam with a score of 700 or higher
 - The exam consists of 250 multiple-choice questions.
 - You have 6 hours to take the exam.
 - The exam is based on 10 domains of knowledge.
- Endorsement
- A current CISSP must sign a form.

There is also an optional audit. To keep a high standard for candidates, ISC² will randomly pick individuals and perform resume reviews to make sure they have the proper level and type of experience to be a CISSP.

Taking the CISSP Exam

- The exam is closed-book.
- Show up early — arrive no later than 8 am. The exam usually starts at 9 am.
- Stay at the hotel the night before.
- Bring a #2 pencil.
- Bring snacks and beverages with a cap.
- Plan for the unexpected.

The following are some tips for taking the CISSP exam:

- The exam is closed-book.
- Show up early—arrive no later than 8 am. The exam usually starts at 9 am.
- Stay at hotel the night before the exam.
- Bring a #2 pencil.
- Bring snacks and beverages with a cap.
- Plan for the unexpected.

Maintaining CISSP

Must stay in "good standing":

- Follow the code of ethics.
- Submit annual maintenance fees.
- Obtain and submit Continuing Professional Education units (CPEs)
 - 120 per renewal cycle

In order to maintain your CISSP you must stay in "good standing," which means you must always follow the code of ethics, submit annual maintenance fees, and obtain and submit the necessary Continuing Professional Education units (CPEs).

CISSP Renewal

- CISSP has a 3 year renewal cycle.
- You must obtain 120 CPE (Continuing Professional Education) credits every 3 years:
 - At least 80 from Group A - directly related
 - A maximum of 40 from Group B - not related
- You must pay the annual maintenance fee of \$85 (US).

The CISSP has a 3 year renewal cycle. You must obtain 120 CPE (Continuing Professional Education) credits every 3 years:

- At least 80 must be from Group A - directly related
- A maximum of 40 may be from Group B- not related

You must pay annual maintenance fee of \$85 (US).

Ten Domains of Knowledge

1. Access Controls
2. Telecommunications and Network Security
3. Information Security and Risk Management
4. Applications Security
5. Cryptography
6. Security Architecture and Design
7. Operations Security
8. Business Continuity (and Disaster Recovery) Planning
9. (Law) Regulations, Compliance (and Investigations)
10. Physical and Environmental Security

Following are the 10 domains of knowledge that are covered on the CISSP exam:

- Access Controls
- Telecommunications and Network Security
- Information Security and Risk Management
- Applications Security
- Cryptography
- Security Architecture and Design
- Operations Security
- Business Continuity (and Disaster Recovery) Planning
- (Law) Regulations, Compliance (and Investigations)
- Physical and Environmental Security

In the following slides, we provide more detail about each domain, and in the next six days, we will cover each domain in depth.

Access Controls

- Access controls
 - Discretionary access control
 - Mandatory access control
- Types of access control
 - Detective
 - Preventive
- Authentication and authorization
- Methods of attack
- Types of access tests
 - Vulnerability assessments
 - Penetration tests

Following are the critical elements that are covered in the Access Controls domain.

These elements are based on the ISC² study guide:

- Accountability
- Access controls
 - Discretionary access control
 - Mandatory access control
- Lattices
- Methods of attack

Telecommunications and Network Security

- OSI and TCP protocol stacks
- Communication media
- Network security
- Firewalls, routers, and network devices
- Development of secure networks

Following are the critical elements that are covered in the Telecommunications and Network Security domain.

These elements are based on the ISC² study guide:

- OSI and TCP protocol stacks
- Communication media
- Network security
- Firewalls, routers, and network devices
- Physical versus logical topologies

Information Security and Risk Management

- Security management concepts
- Non-repudiation
- Security policies and procedures
- Data classification
- Security awareness training
- Outsourcing

Following are the critical elements that are covered in the Information Security and Risk Management domain.

These elements are based on the ISC² study guide:

- Security management concepts
- Non-repudiation
- Security policies and procedures
- Data classification
- Security awareness training
- Outsourcing

Applications Security

- Application development issues
- Agents and applets
- Data warehousing concepts
- System development models

Following are the critical elements that are covered in the Applications Security domain.

These elements are based on the ISC² study guide:

- Application development issues
- Agents and applets
- Data warehousing concepts
- Artificial Intelligence (AI) systems
- System development models

Cryptography

- **Uses of cryptography**
- **Types of cryptography**
 - Symmetric
 - Asymmetric
 - Hash
- **Cryptography architectures**
- **Attacking cryptography**

Following are the critical elements that are covered in the Cryptography domain.

These elements are based on the ISC² study guide:

- Uses of cryptography
- Types of cryptography
 - Symmetric
 - Asymmetric
 - Hash
- Cryptography architectures
- Attacking cryptography

Security Architecture and Design

- Security models
- Security evaluation models
- Computer architecture and security
- Common criteria

Following are the critical elements that are covered in the Security Architecture and Design domain.

These elements are based on the ISC² study guide:

- IP and MAC addresses
- Operating models
- Common criteria
- Security issues
- IPSEC

Operations Security

- Administrative management
- Operation controls
 - Deterrent
 - Preventive
 - Detective
- Auditing
- Monitoring
- Configuration control

Following are the critical elements that are covered in the Operations Security domain.

These items are based on the ISC² study guide:

- Administrative management
- Operation controls
 - Deterrent
 - Preventive
 - Detective
- Auditing
- Monitoring
- Configuration control

Business Continuity Planning

- Business Continuity Planning (BCP)
- Disaster Recovery Planning (DRP)
- Strategies and techniques
- Recovery strategies
- Maintenance of the plan

Following are the critical elements that are covered in the Business Continuity Planning domain.

These elements are based on the ISC² study guide:

- Business Continuity Planning (BCP)
- Disaster Recovery Planning (DRP)
- Strategies and techniques
- Elements of BCP and DRP
- Lifecycle for BCP and DRP

Regulations and Compliance

- Law
- Types of crimes
- Investigations
- Ethics
- Forensics procedures

Following are the critical elements that are covered in the Regulations and Compliance domain.
These elements are based on the ISC² study guide:

- Law
- Types of crimes
- Investigations
- Ethics

Physical and Environmental Security

- Site surveys
- Physical security components
- Technical controls
- Physical security threats
- Components of physical security

Following are the critical elements that are covered in the Physical and Environmental Security domain.

These elements are based on the ISC² study guide:

- Physical security components
- Technical controls
- Physical security threats
- Components of physical security

1. Access Controls

Ten Domains of Knowledge

1. Access Controls

Domain 1 Agenda

- Systems and methodologies
- Terms and principles
- Model
- Measures
- Identity, authentication, and authorization
- Techniques
- Protocols
- Passwords and cracking
- Access control goal
 - Preservation of confidentiality, integrity and availability

Access Controls is one of the most important domains in the CISSP CBK. This domain is closely interlinked to the other domains. As a CISSP, it is extremely important that you master the concepts of access control methodologies, implementation in centralized and decentralized environments, detective and corrective measures, and monitoring. If you fully understand these concepts, you will better understand the potential risks, vulnerabilities, and exposures that involve access control.

Access is the ability to do something with a computer resource (such as use something, change something, or view something).

Access control explicitly enables or restricts access in some way, usually through physical and system-based controls. Computer-based access controls can prescribe not only who or what process may have access to a specific system resource, but also the type of access that is permitted. These controls can be implemented via the computer system or via external devices.

Access Control Systems and Methodology

- Access control is controlling who can do what.
 - Who are the Individuals?
 - What are your resources?
- Access control is concerned with protecting:
 - Confidentiality
 - Integrity
 - Availability
- Access control is concerned with reducing risk.
 - $\text{Risk} = \text{Threat} \times \text{Vulnerability}$

The CIA triad illustrates the basis by which security professionals should assess all of their security efforts. It is often depicted as a triangle with three equal sides. The three sides of the triangle are labeled confidentiality, integrity, and availability. These three properties must be balanced for the type of environment that you maintain. Each of the three properties has its place. You can have an environment with a very high level of confidentiality and integrity in which availability has not been addressed. Although the information might be protected and might not suffer from unauthorized modifications, the information might not be available the moment you need it. Some books refer to CIA in a slightly different order (it's often referred to as AIC), which is essentially in the reverse order of CIA.

Confidentiality is the property that ensures information is not disclosed to unauthorized users. Many believe it is mainly used in a military environment, but it is also commonly used by commercial companies to protect their intellectual property, research data, or other secrets. This gives the company a key commercial advantage. Privacy issues, personal information protection, and legislation have been a driving factor in the increasing importance of confidentiality. In many sectors, it is no longer a "nice-to-have," but a legal requirement.

Integrity is the property that ensures data has not been modified either in transit or while in storage. Depending on the type of transaction, the integrity requirement varies. Listening to streaming media or music over a network link does not require a high level of integrity, even if one packet is dropped or mangled. However, if you have financial data or life-critical data, dropping a single packet could cause major problems.

Availability is the property that ensures access to data when it is needed. It is a frequently neglected item that gets noticed only the day you suffer downtime, which, of course, is always at a critical moment.

Access Control Responsibility

- Data owner
 - Ultimately responsible
 - Final authority
- Data custodian
 - Acts on behalf of the data owner
 - Maintains and administers security

Access control ensures that resources are granted only to those users entitled to them. This section discusses some of the theoretical principles and examples of how they are applied.

The two key roles in controlling access are:

- *Data owner*. The entity (person, system, program, and so on) that has responsibility and authority for the data. Data owners make the decisions.
- *Data custodian*: The entity that currently uses or manipulates the data and therefore temporarily takes responsibility of the data. Data custodians implement the decisions of data owners.

Organizational Control

- Centralized control
 - Single group in charge
 - Could cause delays in response to remote business units
- Decentralized control
 - Controls map to individual business units
 - Causes discrepancies across the organization
 - No consistent view

The two general types of control are:

Centralized Control: With centralized control, it is very clear who is in charge. The benefit is a single view of the organization; but centralized controls can cause delays. The criticism is that centralized controls do not react fast enough to the changing needs of businesses.

Decentralized Control: When you have disparate offices, control cannot be centralized in one location but dispersed throughout the whole organization.

Centralized Access Control

- Remote users
 - Remote authentication and dial-in user service (Radius)
 - Uses authentication server and dynamic passwords
- Callback
 - Remote user dials in to the authentication server (AS).
 - Caller provides ID and password.
 - AS looks up caller's ID in database of authorized users and obtains a phone number at a fixed location.
 - AS calls the phone number; the user answers.
 - User has access to the system.

Remote user access control can be provided with a centralized database that stores user names and user passwords.

Callback can also be used to provide access control. In some callback implementations, the user must enter another password upon receiving a callback. The user has to be at a fixed location whose phone number is known to the authentication server. The threat to the callback system is that an attacker can have the call automatically forwarded to his or her number, enabling access to the system.

Centralized Access Control (2)

- Challenge Handshake Authentication Protocol (CHAP)
 - Central location sends challenge to remote user
 - User responds with encrypted hash of challenge
 - Password not sent in clear over link and messages are encrypted
- Terminal Access Controller Access Control System (TACACS)
 - Requires user ID and static password
- TACACS+
 - Provides better protection
 - Uses tokens for two-factor, dynamic password authentication.

CHAP creates an encrypted hash from the original challenge using the password. The hash is sent to the central authenticator that also knows the password.

Some of the key features of the challenge handshake authentication protocol (CHAP) are:

- The central location sends a challenge to the remote user.
- The user responds with an encrypted hash of the challenge.
- Passwords are not sent in the clear over the link, and messages are encrypted.

The terminal access controller access control system (TACACS) requires a user ID and a static password.

TACACS+ provides better protection than TACACS, and uses tokens for two-factor, dynamic password authentication..

Database Issues

Relational database:

- Relation: two dimensional table
- Rows represent records (tuples)
- Columns represent attributes
- Number of rows: cardinality
- Number of columns: degree
- Domain-set of allowable attribute values

The following are some issues with relational databases:

- Relation: two dimensional table
- Rows represent records (tuples)
- Columns represent attributes
- Number of rows: cardinality
- Number of columns: degree
- Domain-set of allowable attribute values

Database Issues (2)

Personnel relation:

| ss# | SALARY | TITLE | LOCATION |
|-------------|--------|------------|----------|
| 164-49-8863 | \$80k | ENGINEER | BLDG.A |
| 123-45-6789 | \$90k | SECT. MGR. | BLDG. B |
| 187-90-7775 | \$95k | SUPERVISOR | BLDG.C |

In above example, the SS# is the **primary** (unique) key.

In another table or relation, if an attribute has a value matching the primary key in this personnel relation, that attribute is called a *foreign* key.

In this example, if a row designated by a primary key is null, then we say **entity integrity** has been violated.

Referential integrity means that, for any foreign key attribute, the referenced relation must have a tuple with the same value for the primary key. Thus, if in another relation, SS# 164-49-8863 is an attribute and foreign key to the personnel relation, then the personnel relation must have a primary key of 164-49-8863.

Database Issues (3)

Database normalization:

- Removing redundant data
- Eliminating attributes in a table that do not depend on the table's primary key
- Remove repeating groups of data

Database normalization occurs by:

- Removing redundant data
- Eliminating attributes in a table that do not depend on the table's primary key
- Remove repeating groups of data

Controlling Access

- Least Privilege
 - Need to know
- Separation of duties
- Rotation of duties

Over the next several slides we will look at the following critical issues related to controlling access:

- Least Privilege
 - Need to know
- Separation of duties
- Rotation of duties

Least Privilege

- Access control needs good administration.
- Availability versus security: The best security is no availability.
- What is the need of the business?
- Reduce the misuse of privilege.

Access control assumes good management by the administrator. The best security is *no* availability. This is not practical. The worst security offenders seem to come from inside our networks, in the form of the misuse of privilege. If we apply the concept of giving the *least amount of privilege* to fulfill the business requirement, we can reduce misuse of privilege. For example, a customer service representative (CSR) needs access to the customer database. Do they need access to all the records, just a region, just a few fields, or just the current customer who is on the phone? Can we integrate the database and the PBX system to filter the available information for the CSR?

Separation of Duties

- Separation of Duties: Breaking a job into different responsibilities
 - No one person should have total control.
- The more critical the job, the more separation.
 - Example: nuclear weapons
- Separating duties could lead to collusion.

Separation of duties is considered valuable in deterring fraud because fraud can occur if an opportunity exists for collaboration between various job-related capabilities. One of the requirements in separation of duty is that for a specific task requiring a specific series of transactions, there should be no single individual that has the access control permissions to accomplish all of the transactions within the set. A good example of this is a clerk processing claims. This clerk should not have the ability to submit a claim and also authorize its payment.

Separation of duties can be static or dynamic. In today's operating systems, static separation of duties is often implemented through the assignment of roles and allocation of transactions to roles. Dynamic separation of duties is harder to implement and presents a greater challenge. It is often implemented in financial systems to avoid the possibility of conflicts of interest.

The Clark-Wilson access control model that you will see later enforces this principle. Despite separation of duties, fraud can still occur when more than one person controlling a component portion collaborates with others to breach the security of a system. This is known as *collusion*. One attempt at preventing collusion is job rotation.

Rotation of Duties

- For critical functions, the person responsible should be changed on a regular basis.
- Rotation prevents one person from becoming "comfortable" in a position.

Rotation of duties occurs when personnel are moved from one job to another at regular intervals. This helps to detect and minimize fraud. If a user is using the computer resources of company X in a fraudulent way, there is a good chance that this will be discovered by a new employee taking over the job. Another great benefit of rotating employees to different positions within a company is cross-training. Employees are more versatile and can fill in for other employees who might get sick or leave the company.

In real life, job rotation is not a common practice in private companies. This is, however, common in government departments and military environments.

Some companies also force employees to take leave because this can help detect fraud when another employee fills the position while the worker is on leave.

Access Control Model Terminology

- **Subjects: Active**
 - User, process or device
 - Active entity
- **Objects: Passive**
 - Files, directories, pipes, devices, sockets, ports, and so on
 - Passive entity that contains or receives information
- **Rules: Filters**
 - Each rule a security attribute
- **Labels: Sensitivity**
- **Interaction**

The following are the key terms used in access control:

- **Subjects: Active**

A subject is either a user or process. The standard subjects for the operating systems of NDS (Novell), Windows NT 4, and Active Directory (Windows) fall into two major groups: Built-in or User-defined. Built-in subjects can take many forms, but they are defined at the time the operating system is designed. User-defined subjects can encompass the built-in subjects, but administrators tend to use them as the business need arises.

- **Objects: Passive**

An object is a passive entity that contains data. An object can be files, directories, pipes, devices, sockets, ports, and so on.

- **Rules: Filters**

The standard rules for UNIX are Read, Write, and Execute. The standard rules for Windows NT 4 are Read, Write, Execute, and No Access. The standard rules of NDS and Active Directory are more granular. In Windows 2000, there are about 30 rules, which are also known as permissions. Each rule has a positive and a negative.

Labels: Sensitivity

Another set of rules with respect to sensitivity of both object and subject is *labels*. For example, all users will have the label of "clearances" and all data objects will have the label of "classifications." Not all access control systems have labels. Labels (security levels) must not be confused with permissions; they are "in addition" to permissions. Security labels indicate the level of sensitivity of an object. Sometimes there are also sensitivity categories, which are implemented to control the need to know. Although you might have a secret level clearance, it does not mean that you can access all documents that are classified as secret. Labels are introduced when you have Mandatory Access Control, and they are a requirement as soon as there is a classification of B1 or higher within the TCSEC.

The Orange Books clearly states:

"To control access to information stored in a computer, according to the rules of a mandatory security policy, it must be possible to mark every object with a label that reliably identifies the object's sensitivity level (for example, classification), and/or the modes of access accorded those subjects who may potentially access the object.¹¹

- **Interaction**

Each subject is assigned security attributes. Each object is assigned security attributes. These security attributes are the rules. The rules are evaluated in the security reference monitor to allow interaction. The interaction is dictated by policy. Policy is written based on two questions;

- What are the business rules?
- How will the business rules be enforced?

The type of access control system we design determines subject, object, rules, and the interaction of labels. For file systems, there are three primary designs in use today: Mandatory, Discretionary, and Role-based. Each design must use a reference monitor that ensures interactions between the subject and the object are verifiable, tamper-proof, and irrevocable. Each design should be simple and provable.

Access Control Models

- Role-based
 - Capability tables
- Rule-based
- List-based (access control lists)
- Token-based
- Risk matrix

Role-Based Access Control assigns users to roles or groups based on their organizational functions. Groups are assigned authorization to perform functions on certain data.

Ride Set-Based Access Control (RSBAC) targets actions based on rules for *subjects* (entities) operating on *objects* (data or other resources). RSBAC is implemented in a variety of software programs and operating systems (including Linux) and is based on the Generalized Framework for Access Control by Abrams and LaPadula.

List-Based Access Control associates a list of users and their privileges *with each* object. Each object has a default set of privileges that applies to unlisted users.

Token-Based Access Control associates a list of objects and their privileges (called capabilities) *with each* user, the opposite of list-based.

Access Control Types

- Mandatory
- Discretionary
- Non-discretionary
- RBAC

- An Access Control Model is defined as a formal description of a security policy. You might understandably say to yourself, "This does not help me much. What is a security policy?" A security policy captures the security requirements of an enterprise or describes the steps that have to be taken to achieve security. Security models are used in security evaluation, sometimes as proofs of security.
- *Discretionary Access Control* (DAC) consists of something the user can manage, such as a username or password. For example, a user might choose to give a document password to someone without notifying the administrator.

Mandatory Access Control (MAC) controls access by the system. MAC requires a lot of work to maintain because all the data has a *classification* and all users have a *clearance*. Users must have the appropriate clearance to access data classified a certain way. Users cannot give their clearance to another person.

Mandatory Access Control

- Access control is based on information sensitivity such as security labels and data classifications.
- Every entity is given an access level.
 - Entities
 - Users
 - Objects
 - Access level
 - Classification
- A user can only access an object if they have the correct access level.
- Access is enforced by the system and cannot be overridden.

Mandatory access control is strictly enforced by the system and cannot be overridden. Another name for MAC is lattice-based access control (LBAC).

Every entity is given an access level. An entity can consist of either a user or an object. Access is allowed only if the subject has the proper classification to access the object.

Following are the rules that are enforced with this model:

1. Users cannot change security attributes at request.
2. User programs must work within the constraints of access rules.
3. Users can access an object only if they have the correct access level and permissions.

Examples of MAC

- **Linux enhancements**
 - RSBAC by the Adamantix project
 - SE by the NSA
 - LIDS
- **eTrust CA-ACF2 by CA**
- **Multics-based Honeywell**
- SCOMP**
- **Pump**
- **Purple Penelope**

When most people talk about the types of MAC, they refer to models such as Bell-LaPadula Policy and Biba. Implementation examples are Linux enhancements, such as RSBAC by the Adamantix project; SE (Security Enhanced) Linux by the NSA (SE Linux currently supports 30 object classes and 122 object permissions (rules); Flask, which has been incorporated into SE; or LIDS. Each of these must be examined in terms of their ability to allow information flow to/from its associated object class.

The following are additional examples:

- Computer Associate's eTrust CA-ACF2 Security for z/OS and OS/390
- Multics-based Honeywell computer
- SCOMP (Secure Communications Processor)
- Pump (developed by the US Naval Research Laboratory)
- Purple Penelope (An NT workstation MLS wrapper from the British Defense Evaluation and Research Agency)

Following are references for these systems:

- <http://itlab.uta.edu/cse6392/Fall2003/Presentations/Security/Raman%20Security1.pdf>
- http://www.networkpenetration.com/adv_steg_posix_flock.html
- http://www.tresys.com/Downloads/selinux-tools/apol/iflow_help.txt

MAC Strengths

- Controlled by the system and cannot be overridden
- Not subject to user error
- Enforces strict controls on multi-security systems
- Helps prevent information leakage

Mandatory Access Control (MAC) imposes limitations on what a subject can do or what a program can do on behalf of a subject. On high security systems, even the system administrator does not have the ability to make changes at will; these changes are implemented and controlled by the security administrator. This is a great way to prevent a Trojan or another type of malware that runs on behalf of a user.

MAC Weaknesses

- Protects only information in a digital form
- Assumes the following:
 - Trusted users/administrators
 - Proper levels have been applied to an individual
 - Users do not share accounts or access
 - Proper physical security in place

As with everything else in security, MAC is just one small piece of the puzzle. Having MAC properly implemented if you do not have any other supporting elements does not accomplish a secure environment. For example, if you do not have a strong policy, proper screening of your personnel, proper destruction of media, and follow the very basic security principles of need to know and minimum privileges at all times, then you will not be secure.

I often tell students: If you do not have physical security, you do not have security. There have been many cases over the past couple of years in which government information or sensitive financial data was stolen by disgruntled employees or people walking away with the company hard drive after they accessed the data center. The latter scenario is even scarier, as it can allow someone to easily perform identity theft.

Users can be your best allies, or, if they have not received proper training and do not play the role they are supposed to play in helping secure your environment, your worst enemies. You cannot control humans; you can attempt only to modify their behavior through awareness and education.

Discretionary Access Control

- Discretionary
 - Access control lists
 - Tabular listing
 - Designates privileges subjects have to objects
 - Used in local, dynamic environments where there is a need for discretionary assignment of access privileges

In discretionary access control, a subject or other authority has the authority to assign the accessible objects and what privileges are allowed concerning those objects.

Other types of discretionary access control include:

- User-directed—user specifies with limitations
- Identity-based—based only on ID of subjects and objects
- Hybrid—combination of ID-based and user-directed

An access control triple includes:

- Program
- User or subject
- File or object

Discretionary Access Control (2)

- An administrator decides whether a user should have access to an object.
- Control is performed at the discretion of any administrator.
- The owner can change security attributes.

Under this type of control, the owner of a resource determines who has access and what privileges they have.

Under DAC, the policy would be expressed as: There is an owner (a), there is an object (b), there are other users (c).

DAC is, "The owner (a) decides who among users (c) has what level of access on the object (b)."

Examples of DAC

- Windows NT 4.0 and most NIX versions use DAC.
- Windows 2000 can be included as a DAC as long as you talk only about the file and folder permissions.

The Windows platform, used strictly in the workgroup mode of operation, is a good example of DAC. An owner of a file can, at his discretion, grant permission to other users regarding the files he owns. Oftentimes on the Windows platform, you will find enterprise resources, such as policies, files, printers, and other resources that are managed centrally under MAC, but between users and when DAC is used.

Longhorn, the next version of Windows, will make use of a new file system called WinFS. This new version has built-in support for DAC. Following is preliminary documentation from the Longhorn Web site:

"The security descriptor of a "WinFS" item contains a discretionary access control list (DACL) that is managed by the item owner. The DACL is a list of access conditions for individual SIDs. Windows compares these access conditions against SIDs that exist in an authenticated user's access token in order to validate whether any attempted operation by a user on a "WinFS" item is permitted or denied.

"A security descriptor contains a second access control list that is used for Auditing and can only be managed by a Windows system administrator. This is called the system access control list (SACL). It is a list of SIDs for whom any operation performed against the item will be recorded in the Windows Audit Log.

"Windows allows you to construct a discretionary access control list on a "WinFS" item using one or more access control entries (ACEs). Each ACE represents a grant or denial of a certain operation for a certain principal, such as a user or a group.

"Each ACL is therefore an ordered collection of ACEs where each ACE specifies a set of access rights and the SID of a trustee for whom the rights are allowed, denied, or audited. The security policy for a "WinFS" item is completely described by the combination of the DACL and SACL in the security descriptor of the item."

Reference:

More information can be found at: <http://longhorn.msdn.microsoft.com/lhSDK/winfs/daovrwelcometowinfs.aspx>

DAC Strengths

- Convenient
- Easier to use and flexible
- Users feel as if they are in control
- Allows ownership concept
- Simple to understand
- Software personification

DAC offers a great level of flexibility at the object level.

DAC allows the distribution of access control permissions down to the owner of resources.

This is great for small groups of users working together on projects. Just imagine being at work and having to call an administrator every time you have to send a file, attach a document, or any other function that allows you to interact and exchange information with other users. It would quickly become a nightmare for you and for the administrator.

DAC is often combined with mandatory access control whereby both MAC and DAC policies must be satisfied for access to be granted to an object.

DAC Weaknesses

DAC fails to recognize the difference between users and programs.

- Processes are user surrogates and can run arbitrary code.
- Processes can change access control attributes.
- DAC generally assumes a benign software environment.
- DAC is subject to user arbitrary discretion.

Result: Much can happen that is not intended.

- Systems is open to any malicious software (such as Trojans).
- Errors lead to possible great escalation of privilege.
- There is no protection against even "trusted" user error.

DAC is not suitable for an environment in which mandatory access control has to be enforced and strictly controlled.

Within a low security environment where DAC is allowed, you can quickly incur very large administrative overhead while attempting to keep track of permissions that are assigned on-the-fly by users. DAC relies on the end user to ensure no valuable data has been leaked and no compromised software is running on behalf of the user. Because of the lack of centralized control, it might be impossible to keep a strong policy in place.

Non-Discretionary Access Control

Non-discretionary:

- Centrally-administered to meet security policy objectives
 - Role-based: based on roles of individuals in the organization
 - Task-based: based on tasks a subject has to accomplish

There is also a non-discretionary, lattice-based access control that deals with a partially ordered set for which every pair of elements (subject and object) has a greatest lower bound and a least upper bound of access rights.

Role-Based Access Control

- Role-based access control (RBAC)
- Non-discretionary
- Centralized authority
- Database management
- Based on capabilities
- Access rights established for each role

RBAC has emerged as a promising feature of many database management, security management, and network operating system products. The essential advantage of RBAC products is that they allow system administrators to assign individual users into roles. The role identifies users as members of a specific group, based on their capabilities, work requirements, and responsibilities in the organization. Access rights, or security privileges, are then established for each role. A user can belong to multiple roles, which provide the appropriate level of access for their requirements. Thus, the RBAC structure empowers administrators with a tool to regulate which users are given access to certain data or resources without having to explicitly authorize each user to each resource.

Example of RBAC

- Database functionality
 - Adjusting the structure
 - Default sorting order
 - Ability to query
- Microsoft roles
 - Data Reader, Data Writer
 - DENY Data Reader, DENY Data Writer

The easiest of all examples of role-based access control is database functionality. Certain functions, such as adjusting the structure of the database or inserting bulk information into the database should be reserved for a few administrators. A single individual or, arguably, no one should control other functions such as manipulation of the database server default sorting order. Further control, such as the ability to query or the ability to construct ad hoc queries, should be delegated to either groups or be controlled by certain applications.

Microsoft has actual roles in their product named Data Reader, Data Writer, DENY Data Reader, and DENY Data Writer. You can assign an application or a particular screen in an application the capability to look at, but not touch, data that takes two roles. Which roles to require will depend on the structure of your database; it might be as easy as assigning Data Reader, or it might be as complex as assigning DENY Data Writer and Data Reader.

New Implementations of Access Control

- Context-Based Access Control
 - CBAC
 - XML data restrictions
- Privacy-Aware RBAC
 - PARBAC
 - The *purpose* is introduced
- Context-dependent
 - Function of the environment such as location, time of day, etc.
- Content-dependent
 - Function of the information being accessed

Context-based access control means that the access decision will not be based solely on the identity of a subject and what object she tries to access. It will be based on what object the subject is trying to access and the events preceding the access attempt. For example, a user might be limited to 100 connections a day; after 100, she will be denied access. Another example is with the implementation of quota, which is fine for the user to use as the resource as long as her data limit is not exceeded.

The following explains this in more detail:

"Existing access control models developed for XML are extensions of traditional access control lists. These models allow node-level security granularity, by assigning access restrictions to the nodes of an XML document. Users' requests are evaluated based on the users' privileges and access requirements of the requested nodes. However, none of these models support the assignment of access restrictions to data associations or the inferences due to semantics of data."

-Dr. Csilla Farkas and Vaibhav Gowadia

Privacy-Aware RBAC

The PARBAC model is used to enforce privacy policies within an organization.

Access Control Measures

- Types of controls:
 - Preventive
 - Detective
 - Corrective
- Implemented across:
 - Administrative
 - Background checks
 - Policies and procedures
 - Technical
 - Encryption
 - Smart cards
 - Physical
 - Locks
 - Securing laptops
 - Securing magnetic media
 - The protection of cable
- Control combinations
 - Preventive/administrative
 - Preventive/technical
 - Preventive/physical
 - Detective/technical
 - Detective/physical

The access control measures listed could be categorized within one or more of the following categories: Administrative, Technical, or Physical.

Preventive access control systems make it impossible to access resources that the user doesn't have a right to access, such as a locked door or access bits on a file. Detective systems generate alerts when unauthorized access has occurred, but do not stop the access from occurring (for example, a burglar alarm). The last one is a deterrent; it works in conjunction with the preventive access control measures and is a control that doesn't restrict access, but makes it clear that permission to access the resource is denied (for example, a fence or logon banner warning).

Access control types and implementations can be combined. The most popular categories are listed on this slide.

Preventive Controls

- Try to prevent an attack from occurring
- With defense-in-depth, could be partially effective
- Not always effective
- Works with deterrent measures

The preventive controls are most certainly the most important controls. It is always more cost effective to prevent an event from happening than suffering an interruption or disruption and then attempting to recover from that uncomfortable posture. Most of the controls in this category clearly attempt to avoid giving someone the opportunity to commit a crime or compromise a system. This includes security awareness and proper training. A lack of education can generate events that might endanger your security posture.

Oftentimes people do not understand what the difference is between preventive and detective controls. Let's take the logical world as an example. The preventive controls will not allow a user to violate the security policy in place. A deterrent control will present a banner that indicates it is not legal to use a resource unless you are an authorized user, but will not prevent it from happening. In the physical world, doors and locks act as preventive controls, and no trespassing signs act as deterrents.

Detective Controls

- Assumes an attack is successful
- Tries to detect that there is a problem after an attack occurs
- Time critical with detection — an attack is occurring

Detective controls are usually used after the fact. Some of the common detective controls include auditing and Intrusion Detection Systems (IDS). Some of the IDS vendors like to claim real-time detection. However, there is still the need to detect enough packets to recognize a signature or to monitor the data in order to look at behavior. Hiring procedures and human resources policies are two detective controls. By properly validating the references a candidate claims to possess, you can quickly identify that a user has falsified his resume. By rotating positions and forcing users to use their well-deserved leave, you can also discover illegal activities. In the physical world, detective controls are motion sensors, CCTV, or other types of devices that can detect an intrusion taking place or someone trespassing.

Other Controls

- **Deterrent**— Discourages security violations (preventative)
- **Compensating** — Used to provide alternatives to other controls
- **Corrective** — Reacts to an attack and takes corrective action for data recovery
- **Recovery** — Restores the operating state to normal after an attack or system failure

These other controls are not controls in the formal sense; however, they are used in common parlance to describe and categorize data. These are:

- **Deterrent** — Discourages security violations (preventative).
- **Compensating** — Used to provide alternatives to other controls. If there is a weakness in a particular control that you choose to implement, you might want to add another layer. For example, video cameras are great as a preventive control, but a security guard is better. Because a security guard at every door is cost prohibitive, we employ cameras. The compensating control is the camera for the chosen weakness of not having enough guards for the entire location. Both the camera and the security guard are preventive controls but one can compensate for lack of the other.
- **Corrective** — Reacts to an attack and takes corrective action. If a security breach occurs (your preventive control fails), a reset or adjustment needs to take place. If the sensor attached to the server room door is tripped (detective), the alarm should sound (preventive /corrective), alerting the security guard (preventive).
- **Recovery** — Restores the operating state to normal after an attack or system failure. If the sensor attached to the server room door is tripped (detective), the alarm should sound (preventive /corrective) alerting the security guard (compensating). The guard will apprehend the intruder (corrective). The guard will relock the door (recovery).

Control Combinations

- Preventive/administrative
 - Organizational policies and procedures
 - Pre-employment background checks
 - Employment agreements
 - Employee termination procedures
 - Vacation scheduling
 - Labeling of sensitive materials
 - Security awareness training

Examples of preventive/administrative controls are:

- Organizational policies and procedures
- Pre-employment background checks
- Employment agreements
- Employee termination procedures
- Vacation scheduling
- Labeling of sensitive materials
- Security awareness training

Vacation scheduling refers to requiring individuals to take vacations of one week or longer and not a day here and there. If the individual is up to malicious behavior, his or her absence for an extended period makes exposure of the harmful activities more probable.

Control Combinations (2)

- Preventive technical (logical)
 - Protocols
 - Encryption
 - Smart cards
 - Biometrics *{for authentication}*
 - Call-back systems
 - Constrained user interface
 - Database views
 - Mechanism for restricting user access to database information
 - Assembles relationship information from database according to access privileges of user

Some preventive technical controls are:

- Protocols
- Encryption
- Smart cards
- Biometrics *(for authentication)*
- Call-back systems
- Constrained user interface
- Database views

A relation in a relational database can be represented as a table with a unique identifier (primary key) for each entry in the table. A database view can assemble information from multiple tables and present it to the user. However, if the user does not have the proper access privileges for certain data, that data will not be presented.

Control Combinations (3)

- Preventive/physical
 - Fences
 - Man-trap
 - Magnetic card entry systems
 - Biometrics *{for identification}*
 - Guards
 - Environmental control systems (temperature, humidity)

Man-trap (two doors physically separated so that an individual may be "trapped" or contained in the space between the doors).

Biometrics is used for identification in physical controls and for authentication in logical (technical) controls.

These measures also apply to locations providing backup storage.

Control Combinations (4)

- **Detective/technical**
 - Intrusion detection systems
 - Violation reports from audit trail information.
 - Clipping levels can be used to set allowable thresholds on a reported activity.
 - Due to the importance of the audit information, audit records should be protected at the highest level of sensitivity in the system.
- **Detective/physical**
 - Motion detectors
 - Thermal detectors
 - Video cameras

Audit information can indicate variations from "normal" operation.

Clipping levels can be set in order to limit the amount of audit information flagged and reported. For example, a clipping level of 3 can be set for reporting failed logon attempts at a workstation. Thus, 3 or fewer logon attempts by an individual at a workstation are not reported as a violation.

In general, these types of devices require a human to monitor and interpret the resulting information.

Identity and AAA

- Identity is saying who you are.
- AAA
 - Authentication
 - Authorization
 - Accountability

Identity, often referred to as identification, is who you declare to be to a system. The process of *authentication* proves that you are really this person and no one else.

User identification enables *accountability*. It enables you to trace activities to individual users who might be held responsible for their actions. Identity or identification takes the form of Logon (or Login) ID or User ID. Each Logon ID must be unique to the system and every user must use their own ID because sharing IDs destroys user accountability.

After the user has been identified and authenticated, the authorization process will take place. The access control system will look at what access right the user has and make an access decision about allowing the requested operation to take place or not.

Identity

- Identifies who someone is
- Is fairly weak in terms of enforcement
- Broken down into:
 - Positive identification
 - Negative identification

Key criteria:

- Issuing of identity
- Naming standards
- Non-descriptive
- Tracking and auditing
- Unique
- Not shared

Identity in today's modern operating systems most often takes the form of a logon ID. Such IDs are usually very easy to determine because they are based on the first and last name of a person, which gives you half of the secret used for identification purposes. Some systems and applications that are badly programmed will allow for the harvesting of usernames by showing an error message that states "Wrong Username" rather than a generic message that does not indicate if you have a good username or password.

Authentication

- Validates the identity of a user
- Involves a stronger measure than identification
- Usually requires a key piece of information that only the user would know

As mentioned, the authentication process ensures the user really is who he claims to be. There are different authentication factors: something you know, something you have, something you are, and lately we have seen a resurgence of "somewhere you are." All of these factors are covered in more detail later on.

Today, a password is the most common authentication used. Why use something as simple as a password? The easiest answer is that it is cheap and the most commonly supported authentication method by all OSs, applications, and Web sites. Depending on the level of security required, you can also replace the use of a password by a strong authentication mechanism or a one-time password, such as those used by token, OPIE, SKEY, and a few others.

Authorization

- Authorization defines what someone can do once they are authenticated.
- Most systems do a poor job of authorization.
- Authorization is tied closely to the principle of least privilege.

Authorization is defined in TCSEC as an individual's right to use or access an object. Authorization is the process of giving a user credentials or permission after identification and authentication are completed. It will dictate who has access to a specific resource and what a user is allowed to do to specific resources (read, write, and execute). Under today's operating systems, it is usually the system administrator who defines which users are granted access and what they are granted access to.

In the previous paragraph, who has access might include not only users, but also workstations, servers, or specific programs. What a user is allowed to do can include read, write, execute, access directories or files, access libraries, and access servers or other types of resources. A resource can be any of the following: data, programs, printers, tape backup, transactions, servers, and more.

Security Models

- Lattice
- Confidentiality: Bell-LaPadula
- Integrity: Biba
- Commercial: Clark-Wilson

The security models covered in this portion of the course are all significant to the development of a secure computing environment. Some of them were designed for specific environments and do not apply well in today's commercial applications. In the upcoming slides, you will see what each of them does and what areas are addressed by each of them.

You need to remember that each of these models views the world only one way. Biba sees the world as an integrity world, Bell-LaPadula, which is used by the military, sees the world as a confidentiality world. So whenever you look at a model, do not attempt to match it to what you currently use in real life. Most of the operating systems in use today make use of multiple models.

Lattice

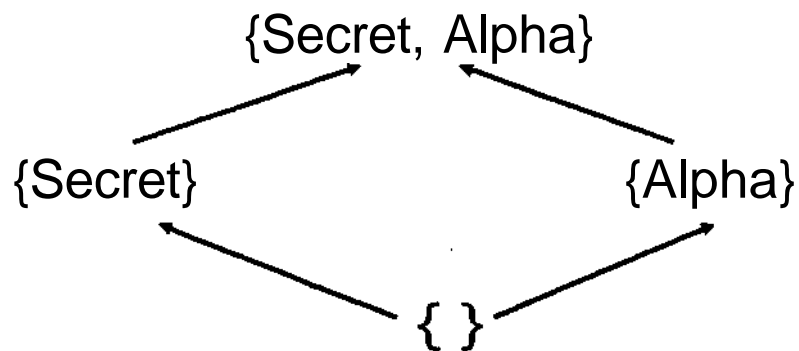
- Deals with information flow
- Formalizes network security models
- Shows how information can or cannot flow
- Drawn as a graph with directed arrows
 - Greatest lower bound
 - Least upper bound

Lattice techniques were developed for situations in which access control must be more restrictive and finer-grained. When talking about lattices, we use the term *object* to refer to files or resources that provide access to information and *subject* to refer to a person or process that accesses objects.

A Lattice model requires that every subject and every object be labeled with one of a number of security designations. Access is granted based on the comparison of those labels; a user of a certain designation can only access resources of the same designation or lower. The United States military has historically used lattice techniques with the designations *top secret*, *secret*, *confidential*, and *unclassified*. So, personnel with *confidential* clearance can access only resources labeled *confidential* or *unclassified*.

The *access matrix* is a similar idea. In the access matrix, rows represent subjects' capabilities, columns represent objects' ACLs, and actions or privileges permitted are listed in each cell.

Example of a Lattice



A lattice is a mathematical construction with:

- A set of elements
- A partial ordering relation
- The property that any two elements must have unique least upper bound and greatest lower bound

A security lattice model combines multilevel and multilateral security. Lattice elements are security labels that consist of a security level and set of categories.

Chinese Wall Model

- Proposed by Brewer and Nash
- Deals with conflict of interest
- No information flow allowed that could cause information leakage that could lead to a conflict of interest (COI)

The Chinese Wall model was proposed by Brewer and Nash to deal with conflict of interest.

No information flow is allowed that could cause information leakage that could lead to a conflict of interest (COI).

Bell-LaPadula (BLP)

- Deals with confidentiality
- Two key principles:
 - No Read Up
 - Obvious for information leakage
 - No Write Down
 - To prevent write-down trojans from de-classifying information

The Bell-LaPadula model is easy to remember. It was written for the military by David Bell and Leonard LaPadula in 1973. This model is mainly concerned with confidentiality and does not address integrity.

There are two main rules with BLP:

- The Simple Security property, which is No Read Up (NRU)
- The * property, which is No Write Down (NWD)

The easy way to remember the rules of BLP is to put yourself in the middle. Take the following example:

TOP SECRET
SECRET—————> This is the level you are at.
CONFIDENTIAL

Being at the secret level, are you allowed to read documents at the top secret level? (No!) (No Read Up)

Being at the secret level, are you allowed to write a secret document into the confidential level? (No!) (No Write Down)

The great innovation here is the No Write Down rule. If you have a Trojan at the secret level, it is not allowed to compromise items at the lower level.

Tip:

There is one BLP property that was not discussed. It is the Strong * property. With the Strong Star property, you are *not* allowed to read down and you are not allowed to write up. You are stuck at a single level, and only within this level are you allowed to perform any operations.

One last word about Tranquility properties:

- *Weak Tranquility property:* Security labels of subjects and objects never change in such a way as to violate a defined security policy.
- *Strong Tranquility property:* Labels never change during system operation.

BIBA

- Deals with integrity
- Opposite of BLP
- Two key principles:
 - No Read Down
 - No Write Up

The BIBA model was developed by Ken Biba in 1975. This model deals primarily with integrity and does not address confidentiality.

The Biba model uses a separate classification system. In contrast to the confidentiality levels used by the Bell-LaPadula model, Biba assigns each user and data resource an integrity level, which might have a similar name (top secret, secret, and so on). Following are the properties of the BIBA . They refer to integrity and are different than the Bell-LaPadula rules:

- *Simple integrity property:* A user cannot write data to a higher integrity level than hers.
- *Integrity star property:* A user cannot read data of a lower integrity level than hers.

The Simple Integrity property protects someone from overwriting data at a higher security level with false information. The Integrity Star property ensures that a user does not receive inaccurate data from a lower level that is less trustworthy.

Let's do the same as we did with BLP. Put yourself in the middle again, but use the BIBA model:

```
GENERAL
CAPTAIN_____> This is the level you are at.
Private
```

Can the captain write an order to the general? The answer is definitively no! So we have the No Write Up rule. Will the captain read (follow) an order from the private? Once again, the answer is no! So we have the No Read Down rule.

As you can see, the rules are similar to BLP, but they are reversed.

Tip: If you take a close look at the CISSP CBK, you will notice that all of the security models that deal with integrity have the letter I in their name, such as Biba, Clark-Wilson, Non-interference, Chinese Wall.

Tip 2: All of the BIBA rules have the word integrity in them. When you see a rule that has the word integrity in it, you will know it is related to BIBA.

Clark-Wilson

- Deals with integrity
 - Unauthorized users cannot make changes.
 - This model maintains internal and external consistency at the system level.
 - Authorized users cannot make unauthorized changes.
- Ensures
 - Internal consistency
 - External consistency
- Integrity enforced through:
 - Well-formed transactions
 - Separation of duties

The Clark-Wilson model emphasizes integrity:

- Internal consistency: Properties of the internal state of a system
- External consistency: Relation of the internal state of a system to the outside world

Mechanisms used for maintaining integrity are well-formed transactions and separation of duties. It is application-oriented, as opposed to a general model such as BLP or BIBA. Subjects and objects are labelled with programs. Programs serve as an intermediate layer between subjects and objects. This model introduces the notion of separation of duties.

Tip: The - (dash) that separates the words "Clark" and "Wilson" should remind you of the separation of duties introduced with this model.

The *Handbook of Information System Management* presents the following definition:

"Wilson and Clark were among the many who had observed by 1987 that academic work on models for access control emphasized data's confidentiality rather than its integrity (the work exhibited greater concern for unauthorized observation than for unauthorized modification). Accordingly, they attempted to redress what they saw as a military view that differed markedly from a commercial one. In fact, however, what they considered a military view was not pervasive in the military."

The Clark-Wilson model consists of subject/program/object triples and rules about data, application programs, and triples. The following sections discuss the triples and rules in more detail.

To ensure that integrity is attained and preserved, Clark and Wilson assert that certain integrity-monitoring and integrity-preserving rules are needed. Integrity-monitoring rules are called certification rules, and integrity-preserving rules are called enforcement rules.

The certification rules address the following notions:

- Constrained data items are consistent.
- Transformational procedures act validly.
- Duties are separated.
- Accesses are logged.
- Unconstrained data items are validated.

The enforcement rules specify how the integrity of constrained data items and triples must be maintained and require that subjects' identities be authenticated, that triples be carefully managed, and that transformational procedures be executed serially, not in parallel.

Access Management

- Account administration
- Maintenance
- Monitoring
- Revocation

Deploying an access control model is only the beginning. User accounts, data, and their relationships must be actively maintained, perhaps by an entire team of employees. This process, called *access management*, consists of four tasks: account administration, maintenance, monitoring, and revocation.

Account administration is just a set of good management practices. The administrator verifies the individual before providing access—*this is the most important step in the process*. This is also an opportunity to teach users not to distribute access privileges they have (tokens, passwords, and so on).

Maintenance is the process of reviewing account data and spot-checking for inconsistencies or errors. Periodically, account staff should review and update lists of users and authorizations.

For accountability, authentications and authorizations should be monitored. System administrators should log both successful and failed attempts.

Almost as important as account administration is *revocation*. Account staff and system administrators should promptly revoke privileges when they are no longer needed, especially for users who have been fired.

Access Control Modes

- Information flow
- State machine
- Non-interference

Often, the authentication and authorization process is said to be in a certain *state*, meaning a user has passed through certain tests that indicate whether she has certain attributes. These attributes are not static attributes, but highly dynamic and relevant only to the current context. For example, as it applies to access control, state usually follows a model similar to:

- *Unauthenticated*: The user has requested access.
- *Authentication pending*: The user has provided a credential.
- *Authenticated*: The credential the user provided is legitimate and valid.
- *Authorization pending*: The user has applied to perform certain functions.
- *Authorized*: The user has legitimate authority to perform the functions requested.

The user can achieve other states if any of the previous tests fail. *The pending* states are typically very short, depending mostly on the processing speed of the authentication and authorization engines.

The Information Flow model attempts to manage access by evaluating the system as a whole. It approaches the system as a system in its current state, which is subject to actions. The subsequent state still should obey the rules for the system as a whole.

Covert channels are the means by which information can flow from a higher to lower classification, despite the explicit rules. The channels can be introduced deliberately to circumvent access controls or can be introduced by programming that transmits information indirectly, without reading the classified file.

Non-interference attempts to track the flow of information rather than manage each discrete access by a subject to an object. It attempts to model access controls around the system behavior as a whole rather than around each action. In this way, it attempts to prevent the information flow associated with covert channels. Unclassified data cannot react with classified data; this would constitute interference.

Information Flow

- How does information flow across a system?
- What changes are made to the information?
- What validation or error-checking is performed?
- Information flow emphasizes the garbage-in, garbage-out principle.

The Information Flow model is closely related to the Lattice approach that we covered at the beginning of this section. It assigns classes that dictate whether an object that is being accessed by a subject can flow into another class.

It is described as the following: A flow is a type of dependency that relates two versions of the same object, and thus the transformation of one state of that object into another, at successive points in time.

In client-server systems, the application programs on the clients manipulate the resources on the servers. Units of resources, such as databases, are named objects. It is significant to consider what subject (S) can access what object (O) by what operation (T) in the access control model.

An access rule is given a tuple (S, O, T). The system is secure if and only if every object is accessed according to the access rules. However, the access control model cannot resolve the containment problem where the information illegally flows among subjects and objects. The Lattice model aims at protecting against illegal information flow among the entities. One security class is given to each entity in the system. A flow relation among the security classes is defined to denote that information in one class (S1) can flow into another class (S2).

In the Mandatory model, the access rule (S, O, T) is specified so that the flow relation between the subject (S) and the object (O) holds. For example, S can read O only if the security class of O can flow to the class of S. Here, only read and write are considered access types of the objects.

In the role-based model, a role is defined in a set of operations on objects. The role represents a function or job in the application. The access rule is defined to bind a subject to the roles.

State Machine

- Based on the processing that occurs, what are the various states a system can be in?
- Each state has a list of prerequisites that must occur.
 - What validation is performed for each state?

The State Machine model will capture the state of a system, and from that point, it will monitor the changes that are introduced either at a specific point or from specific events. The *state* refers to the security features of a system. So when you capture the initial state and the system does some processing or functions, the system should remain in a secure state.

The State Machine model is one of the most abstract models. Even though the aim of the model is to take a snapshot of the security of a system, it was defined without having a definition of security. The model by itself has very little use and you will see that it is combined with other models to give it some meaning.

Non-interference

- There should be no way to predict the output based on variations in the input.
- Each input processing path should be independent and have no internal relationships.

The Non-interference model keeps activities at different security levels separated from each other. The Non-interference model does not allow any flow between the different levels. This is a good way to avoid covert channel communication from taking place where a user modulates resources at one level to send a message to a user at another level where access would otherwise be denied.

Authentication

Authentication is based on:

- Something you **know**
- Something you **have**
- Something you **are**
- **Some place** you are (new)

There are four ways a user can be authenticated:

- Something you know - **password**
- Something you have - **token**, **smart card**
- Something you are - **biometric**
- Some place you are (new) - **location**

Identification is the process of telling the system your logon name/user ID. When you give it a unique secret, you start the *authentication* process. If someone can readily obtain that secret, you have a weak authentication method. If you have secrets in a variety of forms (*factors*), you decrease the likelihood that someone will impersonate you. Users prefer the least invasive form of authentication possible (*user acceptance*). Security professionals prefer the most secure (*biometric*), most accurate (*low CER*) and unique to the individual type of authentication possible (*retina/iris*). The business requirements of authentication are that it be inexpensive, fast, and accurate.

This balance is achieved is two- or three-factor authentication. The factors are, first something you *know* (*password*), second, something you *have* (*token*), third, something you *are* (*biometric*). You can use a biometric device without a token and still have two-factor authentication, but due to the business constraint of cost, you must typically implement tokens before biometrics. As your security needs increase, the number of factors should also.

Something You Know

- Simplest to implement
- Accomplished through passwords
- Easy for user to forget or write down
- Easy for an attacker to guess

The traditional authentication method is something you know, such as a password or passphrase. This has been used since long before computers to prove that one has the right to access a restricted area. The challenge with this type of authentication factor is the human factor. It is necessary to implement a technical means that will ensure the use of strong passwords. A strong password should be constructed, changed, and maintained in accordance with what you are trying to protect. The use of passwords that are *too* strong sometimes decreases the effectiveness of your authentication mechanism because most users *will* write the password down to remember it.

Something You Have

- Accomplished through some form of a token
- Token provides password
- Changes on a regular basis so it is difficult for an attacker to guess
- More expensive to implement because each user needs a token
- Potential for a user to lose token

This form of authentication requires possession of something, such as a key, a smart card, a disk, or another device. Again, this method predates computers by a long time and is used everywhere in the form of door keys, paper documents (certificates, licenses), and so on. One of the factors that makes this difficult to implement is cost. It is necessary to have card readers in some cases or tokens in other cases. These hardware devices have a lifetime and require proper management to ensure their effectiveness.

Something You Are

- Implemented through biometrics
- Very hard for someone to lose
- Does not require the user to have anything
- Each system that authenticates needs a special reader
- Can cause privacy issues

Something you are is really a special case of something you have and has only recently been developed. This encompasses all biometric techniques, such as fingerprint, voice, or retinal scans. This type of authentication has some challenges, such as dealing with the enrollment process, changing health conditions, privacy issues, and attacks.

If you use fingerprints, which are one of the most commonly accepted biometric tools, you have to spend time going through an enrollment process with each of the users. There are also some serious privacy issues attached with some of the biometric devices. There was a case in which a person was able to detect that one of the employee's might be pregnant because of specific changes that showed up in her blood vessel pattern while doing a retinal scan. Last, but not least, some of the biometric devices have been rendered useless by clever and well-planned attacks on them. In one case, there was a security engineer who was able to defeat fingerprint readers eight out of ten times simply by using Jell-O or gelatin to create a copy of the fingerprint.

Some Place You Are

- Based on GPS devices
- Each system needs a device to identify location
- Works well with classified data and controlled access

Access is granted based on the fact you are in a specified location. This tends not to be used except in combination with other authentication methods. A couple of decades ago, when large mainframe computers were the norm, this used to be one of the primary factors in authentication because you had to be physically on the site to use the resource.

Strong Authentication

- Sometimes called two-factor or multi-factor authentication
- Uses two different methods together to authenticate an individual
- What happens if someone loses a Secure ID token?
 - Should also be protected with a password
 - Together this forms two-factor authentication

Successful authentication can demand that two or more of the previous authentication factors be used; this is called strong authentication. "Two-factor" authentication is considered considerably more secure than relying on a single piece of evidence (such as access at an ATM, which requires a PIN [something you know] and a bank card [something you have]). This also mirrors to our concept of defense-in-depth. You should never rely on a single measure to protect your system. When you utilize multiple measures, if one measure is compromised your system will still be secure.

Biometrics

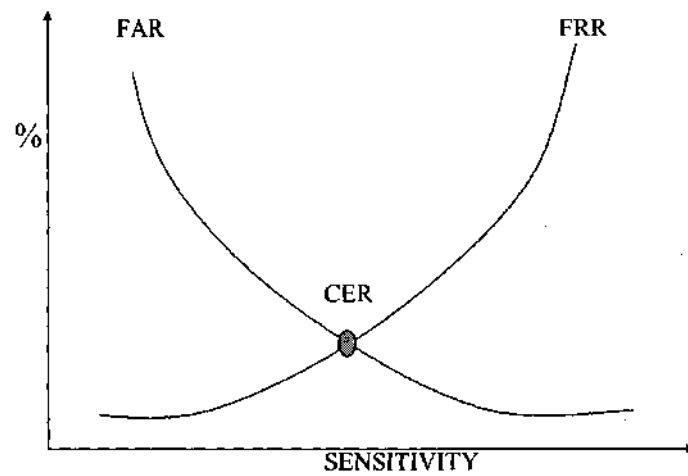
- Used for Identification in physical access control and for authentication logical access control
- Identification
 - "One-to-many" search of an individual's characteristics from a database of stored images
- Authentication
 - "One-to-one" search to verify a claim to an identity made by a person

Biometrics is an automated means of identifying or authenticating the identity of a living person based on physiological or behavioral characteristics.

An example of identification being one-to-many is trying to identify a person by means of fingerprints found at the scene of a crime. A one-to-many search is conducted trying to match the fingerprint with all the possible fingerprints available in a database.

In authentication, it is assumed that a person's fingerprint has been taken and stored in a database under that person's name and ID. Thus, to confirm that the person is who he or she claims to be, one will compare the person's fingerprint with the corresponding fingerprint stored in the database. (One-to-one comparison)

Biometrics



As sensitivity of detecting devices is increased, false rejection rate (FRR) increases.
Conversely, as sensitivity is decreased, false acceptance rate (FAR) increases.
CER is cross over point of the two curves.
The lower the CER, the better the detection device.

Technical Control: Biometric Access Control

- Fingerprint
- Palm scan
- Hand geometry
- Voice print
- Retina pattern
- Iris scan
- Facial recognition
- Keystroke dynamics
- Signature dynamics

Many human features can be used to uniquely identify you. Some features have been around longer than others, but the reason there are many different identification methods is based on reliability, cost, and human factors.

Several of the items on this slide are obvious and self-explanatory, but the following require additional explanation:

- *Hand geometry?* is not hand topology (the side view elevations of parts of the hand), which is not discriminating enough to be effective. Hand geometry includes many characteristics of the hand, such as thickness, width, length, and so on. The palm print, like the fingerprint, is okay.
- *Retina pattern* measures the blood vessels of the eye; it is relatively intrusive.
- *Iris scan* is accomplished by using a camera, perhaps located on the wall, that recognizes an individual's eye(s) as she passes by. This procedure is not intrusive.
- *Facial recognition* matches an individual's facial patterns with the patterns stored in a database.

Technical Control: Biometric Access Control (2)

- Resistance to counterfeiting
- Data storage requirements
- Acceptability to users:
 - Subject/system contact requirements can be intrusive.
 - Enrollment time of 2 minutes per person is standard.
 - Average implementation speed and throughput rate is 6-10 seconds.
- Reliability and accuracy
 - Three unique biometric characteristics are fingerprint, retina, and iris.
 - Crossover error rate must be appropriate to the application.

Unlike other technical controls, because biometrics are tied to an individual, it is more difficult for someone to lose, forget, or give their biometric signature to someone else.

To initially acquire the information can take up to 2 minutes, but because this is done only once, the time investment is acceptable in most situations. After the individual is validated, it takes less than 10 seconds for him to be authenticated. Reliability and accuracy are critical measures when selecting a biometric device.

It is also important to remember that on a computer, a biometric signature is stored as a series of 1s and 0s. Therefore, if that information is not properly protected, it is possible for someone to steal an individual's biometric. If this occurs, it is a critical problem because of the difficulty of having a user change his biometric signature.

Technical Control:

Biometric Access Control (3)

Three main performance measures:

- False reject rate (FRR) (Type I error)
 - Percentage of authentic persons rejected as unidentified/unverified
 - Failure to acquire (sensor not presented with sufficient usable data to make decision)
- False accept rate (FAR) (Type II error)
 - Percentage of unenrolled/impostors accepted as authentic
- Crossover error rate (CER)
 - Percentage at which false rejection and false acceptance are equal
 - The lower the CER, the better the biometric device

When selecting a biometric device you must be aware that many different methods can authenticate an individual. A common means for selecting a method is determining the error rates for a particular method.

A Type I error, often called a false reject rate, is exemplified by a legitimate user of the system being denied access. A Type II error, often called a false accept rate, occurs when an unauthorized person is given access to the system. In most situations, if you had to pick one type of error, you would pick a Type I error over a Type II error because you would rather have authorized people denied access than unauthorized people given access.

After the Type I and Type II errors are determined, you can calculate the crossover error rate. It is when the two values are equal.

Technical Controls: Biometric Access Control (4)

| <i>System Type (CER)</i> | <i>Response Time</i> | <i>Accuracy</i> |
|-------------------------------------|-----------------------------|------------------------|
| Palm scan | 2 - 3 seconds | 0 % |
| Hand geometry | 3 - 5 seconds | 0.1 % |
| Iris scan | 2 - 4 seconds | 0.5% |
| Retina scan | 4 - 7 seconds | 1.5% |
| Fingerprint | 5 - 7 seconds | 5 % |
| Voice pattern | 0 - 14 seconds | 8 % |
| Facial recognition | 2 seconds | TBD |
| Signature dynamics | 5 - 10 seconds | TBD |

Certain practical considerations must be kept in mind when deploying a biometric solution. Accuracy is obviously critical because if the system denies a large amount of legitimate users and a human has to get involved and validate someone, it defeats the purpose of having a biometric system. The second important consideration is the response time it takes to give the user an answer. You can have the best system in the world, but if it takes 30 minutes to give an answer, it is not practical.

This chart shows you the response time and accuracy for several techniques.

| <i>System Type</i> | <i>Response Time</i> | <i>Accuracy (CER)</i> |
|---------------------------|-----------------------------|------------------------------|
| Palm scan | 2 - 3 seconds | 0 % |
| Hand geometry | 3 - 5 seconds | 0.1% |
| Iris scan | 2 - 4 seconds | 0.5% |
| Retina scan | 4 - 7 seconds | 1.5% |
| Fingerprint | 5 - 7 seconds | 5 % |
| Voice pattern | 0 - 14 seconds | 8 % |
| Facial recognition | 2 seconds | TBD |
| Signature dynamics | 5 - 10 seconds | TBD |

Biometrics

- Fingerprints
 - Devices capture "minutiae" of fingerprint
 - Image or details extracted from image compared to data in reference file— finger scan
- Iris scan
 - Passive view of iris by camera
 - Non-intrusive
- Retina scan
 - Laser scan of blood vessel patterns in retina of eye
 - Must press eye up against device
 - Intrusive
 - Blood vessel patterns may also provide information concerning illness
 - Possible exchange of body fluids

In fingerprint systems, the actual fingerprint is stored and requires approximately 250 kb per finger for a high quality image. This level of information is required for one-to-many searches in forensics applications on very large databases. In finger-scan technology, a full fingerprint is not stored, the features extracted from this fingerprint are stored using a small template that requires approximately 500 to 1000 bytes of storage. The original fingerprint cannot be reconstructed from this template. Finger-scan technology is used for one-to-one verification using smaller databases. Updates of the enrollment information may be required because some biometric characteristics, such as voice and signature, may change with time.

An iris scan is performed by remote camera and is non-invasive.

A retinal scan is more intrusive and might be considered uncomfortable.

Health information can be obtained from a retinal scan as a result of observation of blood vessels in the eye.

Possible exchange of body fluids in a retinal scan which might spread infection.

Biometrics Issues

- Key factors in selecting biometrics:
 - Reliability
 - User friendliness
 - Cost
- Additional factors
 - Enrollment time
 - Time to initially "register" by providing samples of the biometric characteristic to be evaluated
 - Acceptable enrollment time is around two minutes
 - Throughput rate
 - Rate at which individuals, once enrolled, can be processed and identified or authenticated by a system
 - Acceptable throughput rates are in the range of 10 subjects per minute
 - Acceptability
 - Privacy
 - Invasiveness
 - Psychological comfort
 - Physical comfort

The key factors in selecting a biometric mechanism are *reliability*, *user acceptance*, and *cost*.

Three quantities typically associated with the reliability of a biometric mechanism are:

- *False acceptance rate* (FAR): The percentage of impostors the biometric mechanism falsely authorizes
- *False reject rate* (FRR): The percentage of legitimate users falsely rejected
- *Cross error rate* (CER) or *equal error rate* (EER): The rate at which the FAR and FRR are equal

Vendors of biometric systems often quote these figures as they pertain to their products. However, the numbers usually pertain to laboratory, rather than real-world conditions.

The user-friendliness of a biometric mechanism is an important factor when choosing one. If people don't like using a system, they will find ways around it. One reason people might dislike a biometric mechanism is if it is *intrusive*. People don't like to touch things other people have touched or to get too close to certain types of machines. For example, they might prefer a voice print identification to a fingerprint or retinal scan mechanism.

The following are key aspects of IDs:

- *False rejects*: People don't like the hassle of using systems that consistently fail to recognize them or require multiple scans to get a match.
- *Enrollment* is the process by which the user's biometric information is initially recorded so it can be used for comparison each time the user tries to gain access. Enrollment should not be difficult, stressful, or time-consuming. If it is, the user will be hesitant to use the system. At the same time, enrollment must sample adequate information to be extremely confident that the person enrolling is authentic and to avoid pushing up the FAR or FRR.

Cost can be the most influential quantity when choosing a biometric system. As you might have figured out, the upfront cost of the system is not the only expense to consider. You need personnel to maintain the system and take on the task of enrollment. In addition, systems that must contact the individual are more susceptible to usage wear and might need to be replaced.

Some technologies are still very expensive (thermograms are 510,000 per scanner). High prices can quickly force a security administrator to examine the current threat model to understand exactly what level of sophistication is needed for access control. Sometimes a conventional lock and key is still sufficient. On the other hand, security personnel should understand that taking custody of sensitive data means making room in the budget for an adequate access control mechanism.

Authentication Centralized Control

- RADIUS
- TACACS
- DIAMETER
- Domains and trusts
- Security domains
- Thin clients
- Constrained user interface

It can get tedious to maintain a large group of services that provide their own authentication and authorization mechanisms (such as PAP or CHAP). One means of simplifying the access control process is to provide a central service that performs the authentication and/or authorization functions. This section gives brief examples of centralized access control implementations.

Dial-In Authentication with RADIUS and TACACS

The *Remote Authentication Dial In User Service* (RADIUS) and the *Terminal Access Controller Access Control System* (TACACS) are protocols that authenticate users on behalf of other services. These services were designed for use with dial-in services, but they can be used for network-based authentication of other services as well.

RADIUS is a UDP-based service and is described in RFCs 2865 and 2866. RADIUS, though not compatible with TACACS, is in many ways viewed as its successor. TACACS is a TCP-based access control protocol described in RFC 1492. It has been in use for decades. Do not confuse TACACS+ with TACACS; the two are not compatible. TACACS+ is another access control protocol that is similar to RADIUS.

Windows NT Authentication

Windows NT uses the notion of *domains* to provide separate authentication zones for an organization or group of organizations. If one department does not want to allow users from another department to access its services, the two departments can use separate domains. Each domain contains individual user accounts, which in turn may belong to one or more groups. In this Role-Based Model, certain privileges might be associated with each group.

Interrelationships between domains are achieved with *trusts*. Useful for sharing resources between organizations, trusts allow the privileges associated with a user in Domain A to be translated appropriately to resources located in Domain B. By placing resources and users in domains, a variety of access rules can be implemented very simply. Domains are nice for users who need only to authenticate once per session to access all of the organization's resources.

RADIUS

- UDP-based
- RADIUS Authentication
 - Access-request
 - Access-accept
 - Access-reject
 - Accounting-request
 - Accounting-response
 - Access-challenge
 - Status-server
 - Status-client
- RADIUS Authorization
- RADIUS Accounting

RADIUS is UDP-based and has the following authentication types

- Access-request
- Access-accept
- Access-reject
- Accounting-request
- Accounting-response
- Access-challenge
- Status-server
- Status-client

TACACS

- TCP-based
- TACACS Authentication
 - Start
 - Continue
 - Reply
- TACACS Authorization
 - Request attribute value pairs (AVPs)
 - Response AVP
- TACACS Accounting
 - Start
 - Stop
 - More
 - Watchdog

TACACS is TCP-based and has the following commands:

- TACACS Authentication
 - Start
 - Continue
 - Reply
- TACACS Authorization
 - Request attribute value pairs (AVPs)
 - Response AVP
- TACACS Accounting
 - Start
 - Stop
 - More
 - Watchdog

DIAMETER

- Draft RFC
- Overcomes limitations of RADIUS
- Authentication
- Authorization
- Accounting
 - Significant improvement

DIAMETER is a draft RFC that overcomes many of the limitations of RADIUS.

Authentication Protocols

- Password Authentication Protocol (PAP)
- Challenge Handshake Authentication Protocol (CHAP)
- Kerberos

This section discusses some popular authentication and authorization protocols. The first two were designed for use with the Point-to-Point Protocol (PPP) and are defined in RFC 1334.

LAN Manager makes use of a hashing algorithm to create an obfuscated version of the password called a "hash." These hashes have numerous vulnerabilities. In some environments, LAN Manager is used without the administrator realizing how weak it is. With a tool, such as RainbowCrack, it is now possible to crack any password up to 14 characters in a matter of seconds.

Both NTLM and NTLM2 are based on hashes. NTLM is susceptible to a DLL injection attack whereby the attacker can force the lsass.exe process into showing all passwords in the weak LM format. These can then be cracked offline by Lophcrack or John the Ripper.

Kerberos has been used for years on UNIX platforms. It is used as the de facto standard in quite a few universities. It has regained a second life with the introduction of Windows 2000 in which Kerberos support is built-in. Like all of the previous issues, there are some security concerns that have to be considered when using Kerberos.

CHAP Versus PAP

- PAP
 - Sends the actual password
 - Vulnerable to a replay attack
- CHAP considered more secure:
 - Password never transverses the network
 - Not vulnerable to a replay attack

Password Authentication Protocol (PAP)

PAP is a simple, weak authentication mechanism. After the user enters her password, it is sent across the network in the clear to the PAP server, where it is validated. Whoever sniffs the network during this transaction can easily discover the password

It is possible for PAP implementations to mitigate the clear-text password risk. The client hashes the password before sending it to the server, where it is compared to a stored hash of the password. If the hashes are identical, the server grants access. This measure really doesn't help much because an attacker can still steal the hash off the network and send it to the PAP server to authenticate (a replay attack).

Challenge-Handshake Authentication Protocol (CHAP)

CHAP should be used instead of PAP whenever possible. It guards against password theft using challenge/response authentication and varies every challenge to prevent replay attacks. A typical CHAP session follows these steps:

1. The client initiates communication with the server.
2. The server sends a challenge back to the client.
3. The user enters his password.
4. The client uses the challenge and the password to create a response.
5. The client transmits the response to the server.
6. The server determines what the response should be using the original challenge and the locally-stored password.
7. If the responses are identical, the server grants access.
8. The server requests re-confirmation with another challenge/response sequence when appropriate.

The main reason to prefer CHAP over PAP is that the user's password, encrypted or not, never traverses the network. It also guards against replay attacks by using an unpredictable challenge every time.

CHAP Steps

1. The client initiates communication with the server.
2. The server sends a challenge back to the client.
3. The user enters his password.
4. The client uses the challenge and the password to create a response.
5. The client transmits the response to the server.

The current slide shows the first part of CHAP authentication. The important thing to remember when you go through the steps is that CHAP does a challenge handshake, which means it is not vulnerable to a session replay attack. If someone replays the same handshake that took place several hours earlier, the user will not authenticate because the challenge will be different each time he tries to authenticate to the server.

CHAP Steps (2)

6. The server determines what the response should be using the original challenge and the locally-stored password.
7. If the responses are identical, the server grants access.
8. The server requests re-confirmation with another challenge/response sequence when appropriate.

Essentially each time the client authenticates to the server the information that is communicated back and forth is different. Both the client and server have a special formula that only they know and therefore even if someone intercepts their communication they will not be able to reproduce the information needed to properly authenticate.

Passwords

- Ideal case — "one-time password"
- Static password
 - Normal passwords with or without expiration time — reusable
 - User-picked
 - System-generated
- Dynamic password
 - Change every time password-generating device is used (one time)
- Account lockout
 - Number of failed attempts
 - Within a certain time period
 - Lock for a specified amount of time

A one-time password is ideal in that if you use it only once and never use it again, it is virtually impossible to steal and use again.

One-time passwords are valid for a given period of time, which could be a minute or any other time interval.

Passwords (2)

- Length range
 - Commensurate with value or sensitivity of resources to be protected
 - Passphrase
 - Sequence of characters that is usually longer than the allotted number for a password
 - Converted into a virtual password by the system
 - Lifetime
 - Shortest practical lifetime
 - Replace when compromise suspected

The length of a password and the frequency of changing a static password are a function of the criticality of the information to be protected.

Many systems will prompt the user and require the user's password to be changed after a specified period of time.

Methods of Password Cracking

- Dictionary attack
- Brute force attack
- Hybrid attack

Many of us remember how, in the movie *WarGames*, a teenager breaks into the government's super-secret WOPR computer by guessing the username and password of the scientist who created the WOPR's software. The teen researched information publicly available about the scientist and guessed that the man's password was the name of his young son, Joshua. That familiar example illustrates exactly why it is important not to use words or names that might be associated with a person. Such information might be readily available to an attacker who could use it to make educated guesses and eventually come up with the right password, even if he does not know the user.

And most of us also are aware that we shouldn't use passwords that are too short (because all the possible character combinations can easily be tried) or write passwords on sticky notes and put them under our keyboards. However, beyond this basic understanding, can we quantify what makes a password difficult to guess when a computer is used as the guessing engine? It depends on the particular method used to protect the sensitive information.

Computers use one-way hashing algorithms to encrypt passwords for storage. A one-way hash is mathematically easy to compute in one direction (for encryption), but nearly impossible to compute the other way, even for computers. This is important because someone who recovers a password file can't use the hashed values to reverse the one-way encryption function and recover the original passwords. How then does the computer use the encrypted information to authenticate users?

The technique is simple. Although hashing functions cannot be reversed, they always produce the same output given the same input. Thus, the computer stores only the hashed passwords (rather than original passwords) on disk. When a user attempts to authenticate to either the machine itself or the network, the computer applies the hash algorithm to the password the user has supplied for authentication. If the hash of the user-supplied password matches the hash stored on disk, the password is correct, and the user is successfully authenticated.

Dictionary Attack

- Easiest and quickest attack to perform
- Not guaranteed to find all passwords
- Relies on the fact that most users pick easy passwords
- Tries every word in the dictionary to see if there is a match

A dictionary attack will use a large file of words as input.

Dictionaries can be used in a cracking program to determine passwords. A short dictionary attack involves trying a list of hundreds or thousands of words that are frequently chosen as passwords against several systems. Although most systems resist such attacks, some do not.

Insiders with expanded privileges can use long dictionary attacks. In this approach, a natural-language dictionary in the native language of the system users is encrypted under the encryption scheme used by the target system. The encrypted values of words in the dictionary are then compared to the encrypted passwords in the password file; a match occurs whenever a password has been chosen from the dictionary.

Three conditions are necessary to the success of a long dictionary attack. First, the attacker must be able to log on to the target system; this condition may be met by the use of a short dictionary attack. Second, the attacker must have read access to the password file. Third, the attacker must know the mechanism and the key variable under which the passwords are encrypted. This condition is often met simply by using the defaults with which the system was shipped. Although these conditions might never be met in a well-managed system, dictionary attacks often work against most systems.

Brute Force Attack

- All passwords are crackable... it is just a matter of time.
- Brute force takes the longest time to perform, but it will find every password.
- Tries every possible combination:
 - **A , AA, AAA, AAB, AAC, and so on**

A brute force attack will try every possible combination of letters and characters that can form a password. Such a process can be very long and it might take days or weeks before you get positive results.

Recently, some very interesting developments have taken place in the password-cracking world. There is a tool called RainbowCrack, which will simply create a database of hashes that include all of the possible passwords that could exist in a character set. If you wish to crack a password, it is simply a matter of querying the database. This usually takes seconds versus hours or days with some of the other tools.

Following is a description of the tool from their main page at <http://www.antsight.com/zsl/rainbowcrack/>:

"In short, the RainbowCrack tool is a hash cracker. Whereas a traditional brute force cracker will try all plaintext possibilities one-by-one in cracking time, RainbowCrack works in another way. It precomputes all possible plaintext-ciphertext pairs in advance and stores them in the file called the "rainbow table." It might take a long time to precompute the tables, but after the one-time precomputation is finished, you can always crack the ciphertext covered by the rainbow tables in seconds."

Hybrid Attack

- Most users append special characters to the end of their passwords.
- Hybrid starts with a dictionary attack and performs a brute force attack of 2-3 characters at the end.

Hybrid attacks are some of the most dreaded attacks. They make use of a dictionary to attempt possible passwords that a user might have picked. For each of the words in the dictionary, there are about 125 variants that will be attempted. John the Ripper, which is a great tool for cracking passwords, has a very flexible language to perform hybrid attacks. It allows you to create your own type of variation.

Tokens

- Supply static and dynamic passwords
- Smart cards
 - Contact
 - Contactless
- One-time passwords (OTP)
 - Counter-based
 - Time-based
- Four types of smart cards:
 - Static password tokens
 - Synchronous dynamic password tokens
 - Asynchronous dynamic password tokens
 - Challenge-response tokens

An ATM card stores information specific to an individual.

Smart cards incorporate additional processing on the card.

There are four types of smart cards:

- Static password tokens
- Synchronous dynamic password tokens
- Asynchronous dynamic password tokens
- Challenge-response tokens

Static Password Tokens

- Owner authenticates himself to the token.
- Token authenticates the owner to an information system.

With a static password token, the owner authenticates to the token. Then, the token verifies the user's pin or password.

The token authenticates the owner to an information system by building a message containing user's name, password, date, and time.

It enciphers this message and sends it to authentication server (AS) and if it can decipher the message properly, authentication succeeds.

Synchronous Dynamic Password Tokens

- The token generates a new, unique password value at fixed time intervals (time of day encrypted with a secret key).
- The password is entered into the workstation along with the owner's pin.
- The authentication system knows the owner's secret key and pin.
- The authenticator verifies that the entered password is valid and that it was entered during the valid time window.

A synchronous dynamic token generates a new unique password value at fixed time intervals. This password could be the time of day encrypted with a secret key. The unique password is entered into a system or workstation along with the owner's pin.

The authentication entity in the workstation knows owner's secret key and pin. The authentication entity verifies that the entered password is valid and that it was entered during the valid time window.

Asynchronous Dynamic Password Tokens

- The tokens are similar to the synchronous dynamic password.
- A new password is generated asynchronously.
- The new password does not have to fit into a time window for authentication.

Asynchronous dynamic password tokens are similar to synchronous dynamic passwords. A new password is generated asynchronously and does not have to fit into a time window for authentication.

Challenge-Response Token

- The workstation generates a random challenge string.
- The owner enters the string into the token along with the proper pin.
- The token generates a response.
- The response is entered into the workstation.
- The authentication mechanism in the workstation determines access.

A challenge-response token works in the following manner:

1. The workstation generates a random challenge string.
2. The owner enters the string into the token along with the proper pin.
3. The token generates a response.
4. The response is entered into the workstation.
5. The authentication mechanism in the workstation determines access.

Single Sign-On (SSO)

- Single sign-on
 - Authenticate once and have access to wide variety of resources
 - Three methods
 - Host-to-host authentication
 - Authentication servers
 - User-to-host authentication
- Can be implemented by
 - Scripts that replay the users' multiple log-ins
 - Authentication servers to verify a user's identity
 - Encrypted authentication tickets to permit access to system services.
 - Kerberos

The idea of single sign-on (SSO) is to eliminate the need to remember multiple passwords when accessing different portions of an information system. SSO is very convenient and time-saving; user does not have to remember many different passwords. The drawback is that once you are in, you are in; so a successful attacker can have access to all the system resources allowed to the user.

In Kerberos-type authentication systems, users are issued temporary authentication tickets to access system resources. Users are given temporary session keys to communicate with different devices on the network. The session keys are only valid for a specified period of time to prevent replay.

Kerberos

- Result of MIT project Athena
- Implemented in Windows 2000
- Kerberos: the name of a 3-headed dog in Greek mythology that guarded the entrance to the underworld
- Third party authentication service
- Provides authentication between subjects and servers providing service to those subjects

Kerberos is a trusted third-party scheme that allows a user to log on to a system once and use any available services without re-authenticating to many different servers. Developed by Massachusetts Institute of Technology's (MIT's) Project Athena, Kerberos is named for the three-headed dog from Greek mythology that guards the entrance of Hades. Kerberos Version 4 was released in 1987 and is still in use in select locations, although Version 5, described in IETF's RFC 1510 and released in 1990, is far more common. Meanwhile, Microsoft incorporated Kerberos in Windows 2000, although that version contains some proprietary extensions to Version 5. While there are significant differences between the three versions, they have many conceptual similarities. Despite its complete approach to providing a cryptographic infrastructure for communications, Kerberos is still far from ubiquitous, mainly because Kerberized applications used to be rare. Now that Windows 2000 has built-in support for Kerberos, the scheme has become much more popular.

Kerberos provides authentication based on secret key technology (DES for V4, DES, Triple DES, or other schemes for V5). Users on the network have conventional passwords that are effectively their secret keys. In addition, every *service* on the network (for example, Telnet, IMAP, etc.) has *its* own secret key, called a *service key*. In Kerberos parlance, we call these services *application services* to differentiate from services offered by the KDC. Instead of users authenticating to services, users and services (also called *principals*) authenticate *to each other*. Servers can detect and thwart imposters, and users can be assured they are not talking to spoofed servers.

Kerberos SSO

Assumptions:

- The trusted key distribution center (KDC) knows the secret keys of all clients and servers on network.
- No passwords or cryptographic keys are passed in the clear on the network.
- KDC is the single point of failure.

The following are some assumptions that Kerberos SSO uses:

- The trusted key distribution center (KDC) knows the secret keys of all clients and servers on network.
- No passwords or cryptographic keys are passed in the clear on the network.
- KDC is the single point of failure.

Kerberos Operation

- The user enters a password into the client workstation.
- The workstation generates the user's secret key, which resides temporarily on the user workstation.
- The client sends the user ID over network to the KDC for an authentication request for a ticket to a service.

When authenticating or encrypting data, both the client and the server must share a secret, randomly-generated session key. As we have discussed, the difficulty with secret key cryptography is in the key exchange. Principals could use their own secret keys to secure the key exchange, but then every principal would have to know every other principal's secret key, which introduces another, much bigger, key exchange problem.

Instead, Kerberos employs a secure, trusted server known as a *Key Distribution Center (KDC)* that stores every principal's secret key. When a new service (such as Telnet or IMAP) is brought online, only the KDC and the new service need to be configured with the service's key, which can be distributed physically or by some other secure means. As Jeff Schiller, who managed this type of system for years at MIT points out, Kerberos is only viable if the KDC is physically secured in such a manner that MIT engineering students cannot pick the lock! KDCs must also be protected at all costs against compromise via the network, lest all the keys stored there be stolen.

If deployed incorrectly, the KDC could be a potential performance bottleneck and a single point of failure. But read-only replicas of the KDC are deployed at various locations in large networks to maximize the availability of remote resources.

Control of the KDC has important political considerations because whoever manages the KDC can access every user's secret key. In large enterprises, it can be politically and practically unacceptable for a single user or team to wield this much power. Typically, this problem is addressed by logically subdividing the enterprise network into distinct *realms*, each with its own independent and locally-managed KDC.

Kerberos Operation (2)

- KDC sends a session key, k_{ctgs} , back to the client encrypted with the user's secret key.
- KDC also sends a ticket-granting ticket (TGT) back to the client.
 - The TGT is encrypted in secret key known only to KDC.
 - It contains time-related information to indicate the freshness of the key and to eliminate replay attacks.

When a participating entity wishes to communicate in a trusted manner with another participating entity over the insecure network, the KDC issues it a ticket that becomes the basis for the establishment of trust between the two participating entities. Kerberos principals use encrypted timestamps to prove to the KDC that they possess the correct secret keys. Hence, the clocks of all participating entities on the network must be synchronized. This is where a service like the network time protocol (NTP) becomes critical.

The KDC comprises two distinct services: the *authentication service* (AS) and the *ticket-granting service* (TGS). In Kerberos terms, *application services* refer to services such as Telnet and IMAP, so they are not confused with the AS and TGS. The steps in establishing an authenticated session between a client and server are:

The Kerberos client software establishes a connection with the AS. The AS authenticates the client and then provides the client with a secret key for this login session (the TGS session key) and a ticket-granting ticket (TGT), which gives the client permission to talk to the TGS. The ticket has a finite lifetime (for example, 10 hours) to reduce the chances of it being stolen while still valid. Because the TGT expires, the authentication process must be repeated periodically if the login session has a long duration.

The client now requests, from the TGS, a *service ticket* for the application service it wants to contact. To make this request, the client must supply the TGS with the TGS session key and TGT it obtained in Step 1. The TGS responds with two copies of a randomly-generated session key: one encrypted with the application service's secret key (this is the *service ticket*) and one encrypted with the client's key. The client can now prove its identity to the application service by supplying the service ticket. The application server responds with encrypted information to authenticate itself to the client. At this point, the client can initiate the intended service requests (for example, Telnet, FTP, HTTP, etc.).

Kerberos Operation (3)

- When client desires service:
 - Client sends a request for service by identifying the service and also sending a time stamp.
 - This request is encrypted with the session key, $k_{c,tgs}$.
 - Client also sends the TGT that was issued to the client earlier by the KDC.
 - KDC sends a session key, k_{cs} to the client encrypted with the client session key, k_{ctgs} .
 - k_{cs} is a common session key between the service and client.

When a participating entity wishes to communicate in a trusted manner with another participating entity over the insecure network, the KDC issues it a ticket that becomes the basis for the establishment of trust between the two participating entities. Kerberos principals use encrypted timestamps to prove to the KDC that they possess the correct secret keys. Hence, the clocks of all participating entities on the network must be synchronized. This is where a service like the network time protocol (NTP) becomes critical.

The KDC comprises two distinct services: the *authentication service* (AS) and the *ticket-granting service* (TGS). In Kerberos terms, *application services* refer to services such as Telnet and IMAP, so they are not confused with the AS and TGS. The steps in establishing an authenticated session between a client and server are:

The Kerberos client software establishes a connection with the AS. The AS authenticates the client and then provides the client with a secret key for this login session (the TGS session key) and a ticket-granting ticket (TGT), which gives the client permission to talk to the TGS. The ticket has a finite lifetime (for example, 10 hours) to reduce the chances of it being stolen while still valid. Because the TGT expires, the authentication process must be repeated periodically if the login session has a long duration.

The client now requests, from the TGS, a *service ticket* for the application service it wants to contact. To make this request, the client must supply the TGS with the TGS session key and TGT it obtained in Step 1. The TGS responds with two copies of a randomly-generated session key: one encrypted with the application service's secret key (this is the *service ticket*) and one encrypted with the client's key. The client can now prove its identity to the application service by supplying the service ticket. The application server responds with encrypted information to authenticate itself to the client. At this point, the client can initiate the intended service requests (for example, Telnet, FTP, HTTP, etc.).

Kerberos Operation (4)

KDC also sends a client-service ticket, to the client encrypted with a key known only to the service. This ticket contains:

- TICKET $t_{c,s}=k_s[c,k_{c,s}]$

When client wishes to use the service, client sends a request for service by sending an authenticator and time stamp encrypted with $k_{c,s}$ to the service.

Client also sends client-service ticket to the service encrypted with key, $k_{c,s}$ known only to service.

Service decrypts the client-service ticket with key it knows.

Service retrieves $k_{c,s}$ from this ticket and then the client and service use this common session key to communicate and implement services.

When a participating entity wishes to communicate in a trusted manner with another participating entity over the insecure network, the KDC issues it a ticket that becomes the basis for the establishment of trust between the two participating entities. Kerberos principals use encrypted timestamps to prove to the KDC that they possess the correct secret keys. Hence, the clocks of all participating entities on the network must be synchronized. This is where a service like the network time protocol (NTP) becomes critical.

The KDC comprises two distinct services: the *authentication service* (AS) and the *ticket-granting service* (TGS). In Kerberos terms, *application services* refer to services such as Telnet and IMAP, so they are not confused with the AS and TGS. The steps in establishing an authenticated session between a client and server are:

The Kerberos client software establishes a connection with the AS. The AS authenticates the client and then provides the client with a secret key for this login session (the TGS session key) and a ticket-granting ticket (TGT), which gives the client permission to talk to the TGS. The ticket has a finite lifetime (for example, 10 hours) to reduce the chances of it being stolen while still valid. Because the TGT expires, the authentication process must be repeated periodically if the login session has a long duration.

The client now requests, from the TGS, a *service ticket* for the application service it wants to contact. To make this request, the client must supply the TGS with the TGS session key and TGT it obtained in Step 1. The TGS responds with two copies of a randomly-generated session key: one encrypted with the application service's secret key (this is the *service ticket*) and one encrypted with the client's key. The client can now prove its identity to the application service by supplying the service ticket. The application server responds with encrypted information to authenticate itself to the client. At this point, the client can initiate the intended service requests (for example, Telnet, FTP, HTTP, etc.).

Kerberos Vulnerabilities

- Kerberos addresses confidentiality and integrity.
- It does not directly address availability.
- Authentication servers are vulnerable to both physical attacks and attacks from malicious code.
- Password guessing can be used to impersonate a client.
- The keys used in the Kerberos exchange are vulnerable:
 - The client's secret key is stored temporarily on the client workstation.
 - Session keys that are stored at the client's computer and at the server can be compromised.

The following are vulnerabilities associated with Kerberos:

- Kerberos addresses confidentiality and integrity.
- It does not directly address availability.
- Authentication servers are vulnerable to both physical attacks and attacks from malicious code.
- Password guessing can be used to impersonate a client.
- The keys used in the Kerberos exchange are vulnerable:
 - The client's secret key is stored temporarily on the client workstation.
 - Session keys that are stored at the client's computer and at the server can be compromised.

SESAME

- Secure European System for Applications in a Multi-Vendor Environment (SESAME)
- Similar to Kerberos
- Distributed access controls with symmetric and asymmetric encryption
- User receives privileged attribute certificate (PAC)

Secure European System for Applications in a Multi-Vendor Environment (SESAME) is similar to Kerberos. It is a distributed access control system with symmetric and asymmetric encryption. The user receives a privileged attribute certificate (PAC).

Prevention versus Detection

Access Control

- Prevention
 - Firewalls
 - Packet filtering
 - Stateful
 - Proxy
- Detection
 - Intrusion Detection System (IDS)
 - Pattern matching
 - Anomaly detection

An ounce of prevention is worth a pound of cure. This is so true in today's information battlefield. Whenever you can avoid an event from taking place, it will cost you about one tenth the cost of recovery. Two well-known and commonly used access control mechanisms are firewalls and Intrusion Detection Systems (IDS).

On the prevention side, there are a few common types of firewalls.

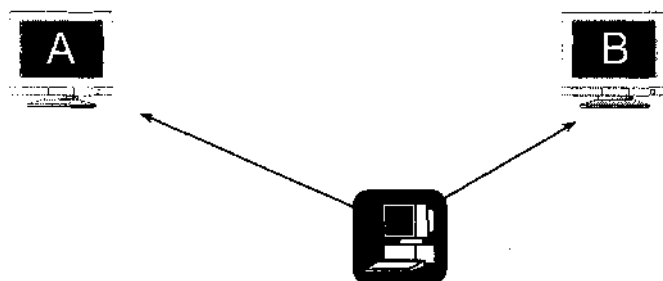
The most basic firewall is a packet-filter firewall. It makes a decision based on IP address and port; it does not keep state information and does not attempt to know if a packet is already part of an established session. They are *not* very smart, but they are fast.

The Stateful concept was invented by Check Point Firewalls. This type of firewall will maintain a state table that keeps tracks of requests being sent outbound and after a reply comes back from the public network, it will validate that the information was requested by an internal user or else the packet will be discarded.

The Proxy firewall has the disadvantage of being slow; however, it is one of the most secure access control mechanisms. There is never a direct connection established between the gateway and an internal user.

IDSs verify, itemize, and characterize the threat from both outside and inside your organization's network, assisting you in making sound decisions regarding your allocation of computer security resources. Using an IDS in this manner is important, as many people mistakenly deny that anyone (outsider or insider) would be interested in breaking into their networks. Furthermore, the information that an IDS gives you regarding the source and nature of attacks allows you to make decisions regarding security strategy driven by demonstrated need, not guesswork or folklore.

Network Vulnerability Scanner



Scanner Warning:

A trusts B

A has potential rshell vulnerability

The best way to protect yourself against these kinds of vulnerabilities is to regularly audit the security of your perimeter. This is a fancy way of saying that you need to hack your own systems. The cardinal rule of scanning or vulnerability assessment is to be certain to scan only systems that you own and are authorized to scan. Otherwise, you will set off someone else's intrusion detection capability and that is never a good idea.

A variety of both commercial and free software scanners exist. If you are shopping for a scanning toolset, it is reasonable to assume that most of the major tools (ISS, NAI, Symantec, Nessus, and so on) scan for the same number of vulnerabilities. They all come up with false positives that have to be investigated manually.

Vulnerability Assessment (VA)

- Scanning key servers looking for a set list of vulnerabilities
- Usually done to look for common or known vulnerabilities
- Performed using a vulnerability scanner tool

A vulnerability assessment or VA occurs when you scan key servers to look for a set list of vulnerabilities. It is usually done to look for common or known vulnerabilities and is usually performed using a vulnerability scanner tool.

There is no point in configuring the scanner to hit all of your addresses unless you are in a small organization — scan one subnet at a time, one workgroup at a time, or whatever makes sense. This way, you won't have an overwhelming number of vulnerabilities to fix.

If you do scan the whole facility, you will get a huge list of problems. With a huge number of problems, people in the organization will talk about fixing them, but because there are so many problems, they likely will not get past the promise stage. This is very dangerous. After you run the scan on a large scale, you will get a huge printout of all the problems. Some of them will be flagged as "very" serious, some will be just somewhat serious, and so on. You present the list to management, tell them it is the end of life as they know it if the problems aren't fixed. Management agrees, they task people, there are meetings, everyone agrees to get things fixed, and then they run into deadlines and emergencies. The problems never get fixed. Now you can't play that card again. After all, the organization is still in business! If you run another scan, no one will take it seriously.

Penetration Tests

- Simulates an attacker trying to break into a network
- Determines whether a site is susceptible to attack
- Are only as good as the person who performs the test
- Do not give a complete view of the security of a network

A penetration test simulates an attacker and tries to break into a network. The goal is to determine whether a site is susceptible to attack. However, this test is only as good as the person who performs the test. It does not give a complete view of the security of a network.

Let's discuss some of the ways a penetration test can be done. A penetration test is sometimes completed at the conclusion of a vulnerability assessment and is used to determine the validity of any identified vulnerabilities. This ensures that all false positives are eliminated if the vulnerabilities are exploited. Penetration tests are sometimes run in lieu of a Vulnerability Assessment and are conducted entirely from outside the network being tested, from the perspective of a true hacker. They can evaluate the effectiveness of your security perimeter, including routers, firewalls, servers, and any other perimeter security devices.

Security Assessment

- More complete view of a company's network security
- Analyzes the entire network from the inside and tries to find the weaknesses
- Offers a complete list of risks against critical assets

A security assessment is a more complete view of a company's network security. It analyzes the entire network from the inside and tries to find the weaknesses. The assessment provides a complete list of risks against critical assets. This type of test is usually recommended if you want to come up with a roadmap and better understand your security risks.

At the conclusion of a security assessment, you get a prioritized list of what the critical risks to your assets are, what the likelihood of the risks occurring are, what the costs are, and what the cost to fix the risks are. Based on this information, management can make the proper decision about what level of risk they are willing to accept. Therefore, a security assessment helps you manage risk in a more "holistic" manner.

Threats

- Malicious code
- Denial of service
- Cramming
- Spamming
- Flooding
- Brute force
- Remote maintenance
- TOC/TOU
- Interrupts
- Code alteration
- Inference

There are various methods to attack a system. In the next few slides, we discuss some of the ways an attacker might remotely penetrate another system.

Although there are probably thousands of different exploits attackers can use against your systems, most can be classified into one or more categories, along with other similar exploits. A lot of research has been done to define a standard vulnerability taxonomy, but so far none has been accepted widely.

Covert Channels

- Covert channels use normal system resources to signal information.
- They cannot be completely removed from a system.
- Two types:
 - Timing channel
 - Using network bandwidth utilization
 - Storage channel
 - Using a hard drive storage

A covert channel is the use of system resources in a way it was not designed for. Some covert channels are based on very accurate timing attacks, while others are based on storage attacks.

An example of a timing attack would be looking at the time required for information to be presented to a user. If the information has recently been accessed and it is still in cache, it will be served to the user requesting it a lot faster than it would be if the system had to read it from the hard disk. A malicious program or user can use this type of timing attack to signal information. If the timing is short, it could mean yes and if the timing is long it could mean no.

An example of a storage attack is a scenario in which a user attempts to access a file. If the file is *not* locked, it can mean yes and if the file is locked, it can mean no.

Covert channel analysis and detection is a requirement of the B2 and higher classification of the TCSEC.

If you are interested in learning more about covert channels, you should visit the following links:

- http://www.sans.org/rr/catindex.php?cat_id=12
- <http://www.cs.ucsb.edu/~kemm/courses/cs177/Covert.pdf>

Malicious Code

- Virus
- Worms
- Logic bombs
- Trojan horses
- Trap doors

In this section, we present our own informal and incomplete list of the types of vulnerabilities we see "in the wild." We do not talk about specific exploits for individual pieces of software, although we might refer to some as examples. We talk about classes of attacks that can be applied to almost any system. Do not think of this as an encyclopedic listing of all possible techniques. Instead, consider it a sampling of sorts. The intent is orientation and education, not enumeration.

Note

As you read through this section, keep in mind that some of the techniques can also be used against attackers, in a devious sort of way. Administrators sometimes intentionally deploy *pseudo flaws*—things that look vulnerable to attack but really act as alarms or trigger automatic actions if an intruder attempts to exploit the "flaw." Do not confuse the single pseudo flaw with the concept of a pseudo flaw extended to encompass an entire host or network that is often referred to as a *honeypot* or a *honeynet*. Neither of these terms properly refers to a single pseudo flaw. The best starting place for more information on this topic is the HoneyNet Project's site at <http://www.honeynet.org/>.

First we look at the common types of malicious code that you will see on a network: virus, worms, logic bombs, Trojan horses, and trap doors.

Man in the Middle Attacks

Masquerading

- Pretend to be a different system
- Active threat

Replay attack

- Play back recorded traffic at a later point in time

Spoofing

- Impersonation
- Active threat

Man in the middle attacks are attacks in which an attacker injects herself in the middle of communication and sees (and possibly manipulates) all traffic going across the wire. The three main types are masquerading, replay attack and spoofing.

Denial of Service (DoS, DDoS)

- A denial of service can occur when a system becomes unusable for various reasons. One example is a system that is unable to respond to new connection requests due to unsolicited connection requests from another system.
- Smurf

Denial of Service (DoS) attacks became quite popular in the last few years, but they have probably been around almost as long as computers themselves. As the name implies, a DoS attack occurs when a user is deprived of the use of data, a computing resource, or service due to malicious actions on the part of an attacker. DoS-like conditions also sometimes occur due to a simple error on someone's part, but they usually aren't referred to as attacks unless the outcome is intentional. There are many different types of DoS attacks available today.

Flooding

- The term flooding can be a misnomer. You might be the victim of a flood; or you might unknowingly be part of the attack.
- Distributed denial of service attacks (DDoS)
- SYN Flood

Flooding attacks are very closely related to resource exhaustion attacks. They involve sending a massive number of requests for a resource or service. The sheer number of requests overwhelms the target and usually makes it either incapable of processing any of them or at least makes the response time to legitimate requests so slow as to be unusable.

One common variation of this attack is the *SYN flood*. Mitnick sent a bunch of SYN packets to Shimomura's server and filled up the request queue, preventing it from responding to anything else until Mitnick had done his dirty work. Fortunately, these sorts of floods are typically easy to pinpoint and defend against because the requests come from a single host, or at least a very small number of them.

You may have heard about a more elaborate (and more effective) type of flood attack known as a *Distributed Denial of Service* or *DDoS*. A DDoS is just like a regular flood, but instead of originating from a single host, DDoS attacks typically come from tens, hundreds, maybe even thousands of hosts all around the Internet. These hosts, known as *zombies*, are so numerous that they can be very difficult to detect and block. If a Web site already experiences a lot of traffic anyway, telling the difference between the legitimate requests and the false ones can be a formidable challenge.

A DDoS involves using a bunch of *slaves* or *zombies*, computers that have been compromised somehow by the attacker and placed at least partially under his control. The zombies usually bide their time and appear to operate normally until the attacker has amassed a large number of them, at which time he will often send a simple trigger that causes them to start their attacks. Individually, none of the zombies cause much real damage. Massed together in a coordinated attack, though, they can be devastating.

The February 2000 DDoS Attacks

Some of the Web's biggest and most high-profile Web sites found out about the effectiveness of DDoS attacks the hard, painful way. On February 7th, 2000, at 10:20 AM PST, the good folks at Web portal giant Yahoo! were no doubt shocked to find their systems flooded with false requests from more than 50 hosts around the Internet. Yahoo!'s US-based services were almost entirely inaccessible until service was restored about three hours later. By one report, 41 percent of their international services were unavailable, making this the biggest DoS attack anyone had seen to date.

Cramming

- The Java Security Manager does not allow port scanning; however an attacker can stick Java code in your browser's cache and have you execute it through a URL.
- Buffer overflows

There are many types of malicious data attacks, most of which rely on poor programming practices with careless error-checking on input data. An attacker might put an alphabetic string where a numerical value is expected, bringing down a popular database application. Worse yet, the attacker might insert UNIX shell commands into input in such a way as to trick the back-end Web application into executing them on the server. These are both common examples of malicious data attacks that could be prevented by more careful validation and checks of input data.

One of the most well-known and popular examples of this sort of attack is the buffer overflow. To communicate with anything, be it a user or another program, an application has to accept some sort of input. It could be a password, a command to be executed, a record from a database, or just about anything else you can imagine. When the program reads this information, where does it go? Even if the information is extremely transient, the application has to store it in memory somewhere, even if it's just for a few clock cycles. This means the program has to set aside a buffer, a region of memory in which to hold the data until it's no longer needed. When the buffer is allocated, the application tells the system how many bytes it will need to store, and the buffer is created to be exactly that size.

Poorly written applications sometimes fail to check that the buffer is large enough to hold all the data it reads. If an attacker is able to provide more data than the application expects, sometimes the attacker can cause the application to accidentally overwrite parts of itself with information supplied in a carefully crafted input string. Usually this causes the application to crash because the attacker can supply extra data the attacker knows is a valid set of machine instructions for that host, tricking the CPU into running code of the attacker's choosing.

This technique of using a buffer overflow to trick a computer into executing arbitrary code is known as *stack smashing* and is quite popular. Although a buffer overflow whose sole purpose is to crash something is definitely a DoS; stack smashing attacks aren't usually considered the same because their main purpose is to run that extra code they include rather than just to render a service unavailable. Still, they use the same basic trick; smashing attacks just take it to a whole new level.

Resource Exhaustion

Type of denial of service attack

Send a large amount of traffic to a device with the goal of overwhelming it so it cannot process legitimate requests

Very difficult to defend against

Resource Exhaustion attacks are pretty high on the list of the easiest attacks to understand and execute. They involve tying up finite resources, making them unavailable to others.

This type of attack can be carried out against almost anything that you can utilize. A classic example is an attack known as *a fork bomb*. UNIX systems use the *fork* system call to create a new process that is an exact clone of the original. Most UNIX systems can handle only a certain number of processes running at one time, although the exact number varies from version to version. The goal of a fork bomb, like the one shown in Figure 1 below, is simply to keep *calling fork* until all possible processes have been utilized. At that point, no one can start any new programs, including the administrator. Because all the standard diagnostic and corrective utilities need to run as separate processes, they effectively become unavailable, and the administrator often ends up hitting the reset switch to reboot the machine.

Figure 1: Fork Bomb Pseudo-Code

```
while (true) {  
    fork() ;  
}
```

Spamming

Spamming is the process of attempting every likely and possible combination of e-mail addresses on a given e-mail server. After compiled, the spamming server sends advertisement e-mails to potential e-mail addresses hoping for hits.

Spamming is the process of attempting every likely and possible combination of e-mail addresses on a given e-mail server. After compiled, the spamming server sends advertisement e-mails to potential e-mail addresses hoping for hits.

Spamming is not only an inconvenience to your users who receive a large amount of unsolicited e-mail, but it can also cause problems for your outgoing e-mail servers. Processing spam can use up the unprotected e-mail server's resources instead of legitimate traffic. In addition, unauthorized e-mail servers that are not supported by the company can also cause additional problems if they are used by people as spam relays.

Brute Force

An attempt to overwhelm a system by bombarding it with possible passwords until the correct one is guessed.

Imagine that you have a very simple electronic keypad lock on a bank vault. Before you can open the door, you have to type in a secret number to pull back the 6-inch (15 cm) diameter deadbolts holding the door closed. If you know the code, there's no problem. You walk up, key it in, and the lock opens, all in a total time of just a few seconds.

What if you do not have the number? No one is likely to tell you the number if you ask. Furthermore, you have absolutely no idea what that security code might look like, except that it's entirely numeric (there aren't any letters on the keypad). You don't even know how long it is. It could be a single digit (we hope not), or it could be 5, 10, 20 digits or longer. To make things more difficult, let's assume perfect physical security of the lock, door, and the vault itself (an impossibility, we know). Opening the lock is the only way in. Just about the only thing you can do at this point is to walk up to the lock and start punching in numbers, hoping you'll eventually hit on the right one.

Congratulations. You just mounted a brute force attack.

Brute force attacks are probably the least sophisticated available. Their goal is to guess a secret of some sort, such as a password or an encryption key. The problem is that they are often nothing more than undirected searches that try every possible combination of inputs until they happen to get lucky. As you can probably imagine, this takes an extremely long time, making it probably the least efficient attack possible.

It's not uncommon for an attacker to connect to a service and repeatedly try to guess valid usernames and passwords until he gets in. This is a classic, but not a very successful brute force attack. Retrying a lot of failed logins takes time, though, and most systems impose a limit on failed login attempts. Some even automatically lock the account associated with the failed attempts, preventing the attacker from trying again.

Remote Maintenance

- Usually creates a backdoor into a system
- Forces a system to use default passwords that others can use to gain access to a system
- Modems: common backdoor created by remote maintenance

For whatever reason, some systems come with well-known account names and passwords already configured and ready for use. Vendors often ship systems in this state, assuming that their customers will change them before they do anything else. Unfortunately, this isn't always the case. Sometimes the new machine's administrator is truly negligent and just doesn't bother, but usually she simply doesn't know that there was something to change. Maybe she knew that her Administrator account needed a password, but she didn't realize that the system shipped with a Guest user as well. Default passwords are quite common, and intruders love to find accounts that still use them. If you spend a little quality time with your favorite search engine, you can uncover extensive lists of these passwords floating around the Internet. Are any of *your* devices listed?

Note

Vendors are catching on to this, but not all of them are fully up to speed. It pays to closely examine all default user accounts on new computers and network equipment as soon as you open the box. If in doubt, call the vendor's technical support line and ask if there are any default passwords installed. If you do not, an attacker might!

One particularly egregious variant of this is vendor maintenance access, often referred to as a *back door* or *trap door*. A back door often takes the form of a special account or maintenance hook allowing the vendor to access the system at a future point. They are usually intended to allow technical support personnel to perform an useful task such as running remote diagnostics or installing upgrades. From a maintenance point of view, this is quite convenient, but the trouble is that you've got to rely not only on your own security practices (which may be quite good) but also on those of your vendor (which may not be).

Time of Check/Time of Use (TOC/TOU)

- Timing is everything
- A TOC/TOU attack simply exploits the difference between when a security control was applied and the time the service was used.
- Race conditions

Race conditions exploit that small window of time between when a security control is applied and when the service is used. These are very tricky and relatively difficult to pull off. To understand why, think of a common example, such as a script that rebuilds access control lists from a database each night to control who gets to access the payroll system.

Note

Race conditions are also sometimes known as *Time of Check/Time of Use* or *TOC/TOU* attacks.

Updating the ACLs nightly is a laudable goal. It keeps the access list updated, removing employees who no longer need access to the data and adding new ones who suddenly do. The list never gets stale because it is completely rewritten each night from scratch. Paradoxically, this actually can cause a problem if the designer of the system is not careful.

One way to update the list would be to remove the existing ACL, and then go through all the entries in the employee database to find out who has the special "payroll access" flag on their account. Those who do are added to the list. After the list is complete, a new ACL is written.

You may have just spotted an obvious point of concern: What happens between the time the ACL is removed and the time when the database search is finished and the new ACL is created? That payroll system is probably wide open for anyone to use. If your employee database is large and takes some time to search, the window of opportunity for an attack could be as much as several seconds, or even a minute or more. That's plenty of time for an attacker to make his own queries while the database is temporarily unprotected.

A slightly more elaborate scheme can hold off deleting the old ACL until the new one is ready to be committed, but even so, this might involve a short window of opportunity for malicious use. Even half a second might be enough, if the attacker can guess when that half a second will occur. Race conditions are sometimes quite subtle and appear in the oddest places. Careful programming and good administration practices can usually clear them up, but you've got to find them first.

Interrupts

- Interrupts are what the computer uses to tell the system that something else needs to be done. What happens when an attacker has compromised a system and uses that interrupt against you?
- Fault line attacks

The next class of attacks doesn't really have a good name as far as we know, so we made one up. We call them *fault line attacks*, comparing them geographic fault lines in which the plates that make up the earth's crust meet. These grind together and fit pretty well, but not perfectly. Sometimes the stresses and movements at fault lines manifest themselves as earthquakes. Similarly, the interfaces between various systems and subsystems in complex environments fit well, but not perfectly. There are a lot of nooks and crannies through which a clever attacker can insert himself and cause the information security equivalent of an earthquake by exploiting gaps in coverage.

Alteration of Code

- Alteration of code is when someone has compromised the *integrity* of your data.
- Rootkits
 - File level
 - Kernel level

Alteration attacks are just what they sound like: Someone makes unauthorized modifications to code or data, attacking its integrity. These attacks can take many different forms and have a variety of consequences. For example, what if an attacker defaced your organization's Web site in a very subtle manner? Instead of posting a trite message about how your site was 0WN3D and that "you suck," maybe they would stealthily modify the numbers in your last quarterly report to make it look like business was very poor to drive your stock price down. Or maybe you caught the latest Web server worm or e-mail virus. (We hope not, but hey, it happens.) Viruses and worms are definitely alteration attacks because they have to modify something on your system to work.

One especially troublesome form of alteration attack is the *rootkit*. A rootkit is a cracker tool meant to be stealthily inserted into the local OS and subverted so that it only does what the attacker wants it to do. This can be a devastatingly effective form of attack. Think about it: Everything you do on a computer has to go through the OS including logging in, tracking down rogue files or processes, and performing forensic analysis. If an attacker installs a rootkit, she has total control over anything the OS says or does. She can grant all her processes automatic root privilege, and there's nothing you can do about it. She can make the OS lie to you and hide her processes and files, so you don't even know you've been attacked. Done well, rootkits are bad news.

Rootkits are usually implemented by means of loadable kernel modules under UNIX, Linux, and NT/2000/XP. Less common (but still effective) is their cousin, interrupt attacks. *Interrupts* are signals that inform the OS that something has occurred. They are usually tied to the hardware in your computer, letting the OS know that a device is ready to read or write some data. For example, if you're using a PC, each key you press generates an interrupt that the system uses to read your input.

The problem with interrupts is that some systems make it possible to install your own interrupt handlers, code that catches the interrupt before the OS does and processes the event first. To continue, with our example, an attacker might install a keystroke logger that watches for keyboard interrupts, writing your keystrokes into a file for later analysis. These loggers usually end up calling the regular keystroke interrupt handler to process the input, so the computer seems to continue working normally. Most people would be none-the-wiser.

Inference

- Inference is the process of determining something you don't know by having details about related items.
- Traffic analysis

An intruder who perpetrates a browse attack simply uses the access she already has to look around and see what's "out there." An attacker might not even need to be a local user, especially in the case where a machine offers information to the world through the Web, FTP, or other public Internet services.

You might think that browsing doesn't really count as an attack because it doesn't involve much in the way of technical knowledge or skill. We hope to convince you otherwise. There's a lot for an intruder to see on a typical system. If they are really lucky, they might find your company's secret business plans or the name and phone number of the CEO; but even less obvious and critical information can be useful.

Most Windows machines will let you browse the network to discover file servers, domain controllers, and printers you might be able to access. Under UNIX, normal user accounts can usually provide a lot of information about printers, file servers, and NIS servers. An attacker might be able to use all this information to map out possible trust relationships. In fact, the first phase of Mitnick's attack began with simple browsing to see what information Shimomura's systems were willing to give away for free.

Emanations

- Information leaving a system
- EMI (electronic magnetic interference)
- Protected with TEMPEST
- Similar to virtual shoulder surfing
- More preeminent with older computers

In order to defeat the problem of compromising emanations the United States Government established the TEMPEST program. Begun in the mid 1950's, this program focuses on evaluating and screening companies and equipment to ensure that electromagnetic radiation from information-handling devices are eliminated or controlled. Any Classified Information Processing System (CLIPS) can emit Compromising Emanations (CE). Regardless of the type, any ordinary electric typewriter or a large data processor emits CE's. The study of the nature of the CE's is referred to as TEMPEST. Foreign governments continually engage in attacks against U.S. secure communications and information processing facilities for the sole purpose of exploiting CE.

<http://www.cs.nps.navy.mil/curricula/tracks/security/AISGuide/navchl6.txt>

Browsing

This is probably the simplest attack to do. It involves looking at large available amounts of data to find compromising information.

Note

Browse attacks are especially devastating when used by insiders who abuse the legitimate access they already have. Not only do they typically have more time in which to browse unhindered and undetected, but your own users are the ones best able to understand the significance of information they find. It has often been said that 80 percent or more of successful attacks come from insiders; and browsing attacks account for a large number of these.

Browsing attacks can be even more dangerous when attackers can *infer* or make connections between different pieces of known information to arrive at a conclusion about other protected information. For example, Mitnick used [^]wger to find out which users were logging in to which computers. From this, he was able to notice patterns of repeated usage and infer that a trust relationship might exist between the terminal and its server. You should see that there is a very close relationship between inference and browsing.

Buffer Overflows

- Put more data into a memory location than what was allocated
- Four conditions needed:
 - Overflow condition
 - Put exploit on memory stack
 - Overwrite return pointer
 - Set return pointer to exploit code
- Ping of Death

There are many types of malicious data attacks, most of which rely on poor programming practices with careless error-checking on input data. An attacker might put an alphabetic string where a numerical value is expected, bringing down a popular database application. Worse yet, he might insert UNIX shell commands into input in such a way as to trick the back-end Web application into executing them on the server. These are both common examples of malicious data attacks that could be prevented by more careful validation and checking of input data.

One of the most well-known and popular examples of this sort of attack is the buffer overflow. In order to communicate with anything, be it a user or another program, an application has to accept some sort of input. It could be a password, a command to be executed, a record from a database, or just about anything else you can imagine. When the program reads this information, where does it go? Even if the information is extremely transient, the application has to store it in memory somewhere, even if just for a few clock cycles. That means the program has to set aside a buffer, a region of memory in which to hold the data until it's no longer needed. When the buffer is allocated, the application tells the system how many bytes it will need to store, and the buffer is created to be exactly that size.

Dumpster Diving

- Extracting information from what people throw away
- Defeated with shredding

Dumpster diving is a nice way of saying you are looking through garbage. However, remember that people throw out some very sensitive information.

Traffic Analysis

- Reviewing the logs of all your security components on your network can help determine where your weaknesses are and who may have slipped under the wire.
- Passive threat

Traffic Analysis is a special type of inference that looks at communication patterns between entities in a system. Knowing who's talking to whom, when, and for how long can sometimes clue an attacker in to information of which you'd rather she not be aware.

This might seem like a rather simple attack, but it can be very effective. By understanding pattern flows across a network, attackers can morph their traffic to look like legitimate traffic and reduce the chances of being detected. If your network has a large amount of Type A traffic and an attacker sends network Type B traffic, the attacker is easy to detect. However, if the attacker changes the traffic to look like Type A, he will better blend into your existing network and be much harder to detect.

Threats to Access Control

- User distrust of biometrics
- Misuse of privilege
- Poor administration knowledge

User distrust for biometrics is not something new and has always been a threat to access control. If a user does not feel good about using an invasive system, such as biometrics, he is simply going to find a way of *not* using them. Some users see biometrics as an invasion of privacy whereby their very own physical characteristics are read and put on file. They are scared of the consequence of having a record on a server.

It is nice to look at the order of acceptance of the different biometric mechanisms. The order of acceptance has slightly changed in the past years. Three years ago, Iris was the most accepted method; however, today voice pattern is by far the most accepted. Here is the list from most accepted to least accepted:

- Voice pattern
- Keystroke pattern
- Signature
- Hand geometry
- Handprint
- Fingerprint
- Iris
- Retina pattern

Another common threat to access control is the misuse of privilege. This misuse could be from an external source or it could very well be one of your employees. Abuse from internal users is always more likely to succeed because they have an intimate knowledge of the security mechanisms in place and what to do to bypass or disable them.

The last, but *not* the least, threat to access control is the introduction of complexity within our environments. Our security architectures are now so complex that we cannot make sense of them. Some of the components of the infrastructure that are not well-maintained will eventually become a threat to your environment if they are not properly configured, updated, and managed.

Current Practices

- Implement MAC if possible
- Use third-party tools in RBAC for NDS and AD
- Layered defenses
- Tokens
- Biometrics

As you have probably realized, there are more passwords today than ever before. Passwords are used to access our online banking, our ADSL and modem links, and corporate e-mail or accounts. This collection of passwords can quickly become unmanageable.

To simplify your life, you can attempt to use Single Sign-On, whereby you authenticate only once in the morning and credentials are carried around to each of the servers in your environment. If your budget is not limited, you can also consider some of the leading token manufacturers, such as SecureID or CryptoCard. They have strong authentication solutions for a fair price.

Last but not least, is biometrics. This is one of the strongest means of authentication if managed properly. Do make use of it if possible.

Domain 1: Summary

Identification

Authentication

Authorization

How attackers compromise these measures

It is critical that whenever you look at a technology or domain, you look at what it is and you also look at how attackers break it. In this section, we looked at the how to validate an individual and how to control access. No matter how strong a security method is, it can always be defeated. In many cases, the weakest link might not be an outsider, but a trusted insider.