ISS

PDFs

Lectures:

1. CHAPTER 1 Accountability and Access Control .pdf
2. CHAPTER-2-Attacks-and-Monitoring.pdf
3. CHAPTER-3-Security-Management-Concepts-and-Principles.pdf
4. CHAPTER-4-Data-and-Application-Security-Issues-Part-1.pdf
5. CHAPTER-4-Data-and-Application-Security-Issues-Part-2.pdf

Sections:

1. Section 1.pdf
2. Section 2.pdf
3. Advanced Encryption Standard-1.pdf

Chapter 1: Accountability and Access Control

I. Core Concepts & Security Principles

Access Control Overview

- **Access controls**: Security features regulating how users/systems interact with resources
- **Access**: Flow of information between a subject and an object
- **Subject**: Active entity (user, program, process) that requests access to accomplish a task
- **Object**: Passive entity containing information (computer, database, file, field, directory, program)
- Access controls enable organizations to control, restrict, monitor, and protect resource availability, integrity, and confidentiality

The CIA Triad

Every security control provides at least one of these principles:

1. **Availability**: Resources must be accessible timely so productivity is not affected
   - Fault tolerance and recovery mechanisms ensure continuity
   - Example: Stockbroker needs accurate, timely information for trading
   - User productivity greatly affected when data is inaccessible
2. **Integrity**: Data must be accurate, complete, and protected from unauthorized modification (**no unauthorized write**)
   - Security mechanisms must alert users if illegitimate modification occurs
   - Example: Intercepted diplomatic email being altered could have severe consequences
3. **Confidentiality**: Information not disclosed to unauthorized entities (**no unauthorized read**)
   - Examples: Health records, financial accounts, criminal records, source code, trade secrets, military plans
   - Mechanisms: Encryption, logical/physical access controls, transmission protocols, database views, controlled traffic flow
   - Example: Soft drink company protecting secret recipe

Important: Different security mechanisms provide different degrees of availability, integrity, and confidentiality. Environment, data classification, and security goals must be evaluated before purchasing security products.

II. The Four Steps of Access

For a subject to access an object, four steps must occur **in order**:

1. Identification

Definition: Method of ensuring a subject is the entity it claims to be

Identification Component Requirements:

- Each value must be **unique** (for user accountability)
- Follow a **standard naming scheme**
- Be **nondescriptive** of user's position or tasks
- **Never be shared** between users

Methods: Username, account number, smart card swipe, token device, spoken phrase, biometric scan

Status: Identity is considered **public information**

2. Authentication

Definition: Verification of identity by comparing credential factors against stored information

Authentication is private information - ability to maintain secrecy directly reflects system security level

Authentication Factor Types

Type 1 (Something you know): Any memorized string of characters

- Password, PIN, passphrase, mother's maiden name, favorite color
- **Weakest** but most common

Type 2 (Something you have): Physical device you possess

- Smart card, token device, memory card, USB drive
- Includes "somewhere you are" (physical location)
- **More secure than Type 1**

Type 3 (Something you are): Body part or physical characteristic (biometric)

- Fingerprints, voice prints, retina patterns, iris patterns, face shapes, palm topology, hand geometry
- **Most secure**

Additional Factors:

- **Something you do**: Writing signature, typing passphrase (keyboard dynamics), speaking phrase
  - Often included in Type 3 category

- **Somewhere you are**: Computer terminal, phone number (caller ID), country (IP address)

Authentication Strength

- **Two-Factor Authentication**: Requires two **different types** of factors (e.g., Token + PIN)
  - Example: Driver's license (Type 2) + phone number (Type 1) when cashing checks
- **Strong Authentication**: Two or more factors (can be same type)
- **General Rule**: Combining different factor types creates more secure authentication

Critical: Identification and authentication are **always together** as a single two-step process. Neither element alone is useful.

3. Authorization

Definition: Ensures requested activity is possible given the rights/privileges assigned to the authenticated identity

Process:

- System evaluates an **Access Control Matrix** comparing:
  - Subject
  - Object
  - Intended activity
- If specific action is allowed → subject is authorized
- If specific action is not allowed → subject is not authorized

Critical Distinction: Just because a subject is identified and authenticated does NOT automatically mean it is authorized

Authorization indicates who is trusted to perform specific operations

4. Accountability (Auditing)

Definition: Process of tracking and recording online activities of user accounts and processes

Purpose:

- Produces **audit trails**
- Reconstruct events
- Verify if security policy or authorization was violated

NIST "Minimum Security Requirements" (MSR) for Auditing (NISTIR 5153)

Audit data recording must comply with **13 requirements**:

1. Provide mechanism for security audit trail supporting after-the-fact investigation
2. Provide end-to-end user accountability for all security-relevant events
3. Protect security audit trail from unauthorized access
4. Provide mechanism to dynamically control recorded event types during normal operation
5. Protect audit control mechanisms from unauthorized access
6. By default, write records for numerous specific security-related events
7. Provide privileged mechanism to enable/disable recording of other events
8. Each audit record must identify several specific data points minimum
9. **Password character strings must NEVER be recorded** in security audit trail
10. Provide option to enable/disable recording of invalid user IDs during failed authentication attempts
11. **Audit control data must survive system restarts**
12. Provide mechanism for automatically copying audit trail files to alternative storage after customer-specifiable time period
13. By default, write records for numerous specific security-related events (reiterated)

Logical Access Controls

- Tools used for identification, authentication, authorization, and accountability
- Software components enforcing access control measures for systems, programs, processes, and information

- Can be embedded in: Operating systems, applications, add-on security packages, database/telecommunication management systems

---

III. Identification & Authentication Techniques

1. Passwords (Type 1)

Why Passwords Are Weak (7 Reasons)

1. Users choose passwords that are easy to remember → easy to guess/crack
2. Randomly generated passwords are hard to remember → users write them down
3. Passwords easily shared, written down, and forgotten
4. Passwords stolen through observation, recording/playback, security database theft
5. Often transmitted in clear text or with easily broken encryption
6. Password databases often stored in publicly accessible online locations
7. Short passwords discovered quickly in brute-force attacks

Password Types

Static Passwords: Always remain the same

Dynamic Passwords: Change after specified interval of time or use

One-Time Passwords (OTP): Change every time they are used

- Considered **strongest password type** (in concept)
- Humans can't remember infinite series of lengthy random strings
- Often implemented as **Type 2 factors** using a token device

Composition Password: Two or more unrelated words joined with number/symbol

- Easy for computers to generate

- Should not be used for extended periods (vulnerable to guessing attacks)
- If generation algorithm discovered, all passwords compromised

Passphrase: String of characters much longer than password

- System converts it to virtual password for authentication
- Often modified natural-language sentences for easy memorization
- Example: "She *ell* C shells ByE the c-shor"
- **Benefits**: Difficult to brute-force crack, encourages many characters, easy to remember

Cognitive Password: Series of questions about facts/predefined responses only subject should know

- 3-5 questions asked (e.g., birth date, mother's maiden name, division manager, exam score, favorite player)
- Most effective systems ask different questions each time
- **Primary limitation**: Each question answered at enrollment AND logon (increases time)
- Often used by financial organizations for phone/web authentication
- **Considered inappropriate and insecure for protecting IT**

Password Policies & Restrictions

Common restrictions:

- Minimum length required
- Minimum age required
- Maximum age required
- Require 3-4 character types (uppercase, lowercase, numbers, symbols)
- Prevent password reuse

Password Change Frequency: As importance of security increases, passwords must change more frequently. Longer static periods and more frequent use = higher compromise likelihood.

Strong Password Guidelines

Security policy must clearly define need for strong passwords and what constitutes one. Users need training to respect and adhere to policy.

DON'Ts:

- Don't reuse any part of: name, logon name, email, employee number, SSN, phone number, extension, identifying codes
- Don't use dictionary words, slang, or industry acronyms

DOs:

- Do use nonstandard capitalization and spelling
- Do switch letters and replace letters with numbers

Password Attacks & Security Measures

Attack Types:

1. **Network Traffic Analysis (Sniffing)**: Capturing network traffic during password entry
   - Attacker attempts to replay packet containing password
2. **Password File Access**: Gaining access to password database file
   - File copied and password-cracking tool used to extract usernames/passwords
   - **Only read access required** (not write access)
   - Makes finding weak user account more attractive than attacking admin/root directly
3. **Dictionary Attack**: Script of common passwords and dictionary words attempts to discover password
4. **Brute-Force Attack**: Systematic trial of all possible character combinations
5. **Hybrid Attack**: Dictionary attack followed by brute-force modifications
   - Adds prefix/suffix characters to dictionary passwords
   - Discovers "one-upped-constructed passwords" (password differs by single character from dictionary form)
   - Examples: "password1", "Password", "1password", "passXword" are all one-upped from "password"

6. **Social-Engineering Attack**: Deceiving user (usually over telephone) to perform system actions
   - Example: Changing executive's password "who is on the road"
   - Example: Creating fictitious employee account

Security Improvements:

Account Lockout:

- Disables user account after specified number of failed logons
- Stops brute-force and dictionary attacks against logon prompt
- After lockout, displays message with time, date, location of last successful/failed logon
- Users suspecting attack can report to administrator
- Auditing tracks logon success/failure
- IDS can identify logon prompt attacks and notify admins

Other Security Measures:

1. Use strongest one-way encryption available for password storage; never transmit over network in clear text or weak encryption
2. Use password verification/cracking tools against your own database; require weak/discovered passwords be changed
3. Disable accounts for short inactivity periods (week or month)
4. Delete accounts no longer used
5. Properly train users about security necessity and strong passwords
6. Require users change passwords consistently (more secure/sensitive environment = more frequent changes)
7. Never display passwords in clear form on any screen or form
8. **Longer passwords (16+ characters)** harder for brute-force tools but harder to remember (leads to writing down)
9. **Security awareness rule: No passwords should ever be written down**
   - **Exception**: Long, complex passwords for most sensitive accounts (administrator/root) can be written down and stored in **vault or safety deposit box**
10. Create lists of passwords users should avoid

11. If root/administrator password compromised, change **every password on every account** (high-security: format drives and rebuild entire system)
12. **Never transmit passwords via email**

---

2. Biometrics (Type 3)

Definition: Behavioral or physiological characteristic unique to a subject

Types: Fingerprints, face scans, iris scans, retina scans, palm scans (palm topography/geography), hand geometry, heart/pulse patterns, voice patterns, signature dynamics, keystroke patterns (keystroke dynamics)

How Biometrics Work

- Biometric data turned into binary data
- Compared for identity validation
- Can be used for **identification** (one-to-many search) or **authentication** (one-to-one match)
- For identification: Used in **physical access controls**
- For authentication: Used in **logical access controls**

Critical Requirement: Must be extremely sensitive and accurate

Biometric Error Rates

Type 1 Error (False Rejection Rate - FRR):

- Occurs when device is **too sensitive**
- Valid subject is NOT authenticated
- Ratio of Type 1 errors to valid authentications

Type 2 Error (False Acceptance Rate - FAR):

- Occurs when device is **not sensitive enough**
- Invalid subject IS authenticated
- Ratio of Type 2 errors to valid authentications

Crossover Error Rate (CER) / Equal Error Rate (ERR):

- Point where FRR = FAR
- Used interchangeably
- **Single purpose**: Compare accuracy of similar biometric devices from different vendors/models
- **Device with lowest CER is most accurate overall**
- Biometric devices have sensitivity adjustment to tune FRR/FAR

Biometric Operational Metrics

Enrollment Time:

- Time required to scan and store biometric factor (create reference profile/template)
- Varies greatly by characteristic type
- **Acceptable**: Under 2 minutes
- Longer enrollment = less user acceptance
- **Reenrollment required** at regular intervals for characteristics that change (voice tones, facial hair, signature pattern)

Throughput Rate:

- Time system requires to scan and process subjects after enrollment
- More complex/detailed characteristic = longer processing
- **Acceptable**: 6 seconds or faster

Acceptance:

- Depends on subjective perceptions: privacy, invasiveness, psychological/physical discomfort
- Health concerns about scanning devices or transferring body fluids

Specific Biometric Technologies

Fingerprints:

- Uses macroscopic patterns on last digit of fingers/thumbs
- **Minutia matching**: Examines microscopic view of fingertips
- **Affected by**: Temperature, pressure, minor surface damage (rough surface sliding)

Face Scans:

- Uses geometric patterns of faces
- **Technology**: Eigenfeatures (facial metrics) or eigenfaces
- "Eigen" = German word for recursive mathematics analyzing intrinsic/unique numerical characteristics

Iris Scans:

- Focuses on colored area around pupil
- **Second most accurate** form of biometric authentication
- **Cannot differentiate identical twins**
- **Longest useful authentication lifespan** (iris remains unchanged throughout life except for eye damage/illness)
- **Acceptable to users**: No direct contact, doesn't reveal personal medical information

Retina Scans:

- Focuses on blood vessel pattern at back of eye
- **Most accurate** biometric authentication (differentiates identical twins)
- **Least acceptable**: Reveals medical conditions (high blood pressure, pregnancy)
- Requires eye placed on cup reader that **blows air into eye**

Palm Scans (Palm Topography/Geography):

- Uses whole hand area (palm and fingers)
- Functions as hand-sized fingerprint
- Analyzes grooves, ridges, creases, and fingerprints

Hand Geometry:

- Recognizes physical dimensions (width/length of palm and fingers)
- Can be mechanical or image-edge (visual silhouette) graphical solution

Heart/Pulse Patterns:

- Measures pulse or heartbeat
- Ensures real person is providing biometric factor
- Often employed as **secondary biometric** to support other types

Voice Pattern Recognition:

- Relies on sound of subject's speaking voice
- **Different from speech recognition** (which extracts communications/automatic dictation)
- Voice pattern: Differentiates between one person's voice and another
- Speech recognition: Differentiates between words within any person's voice

Signature Dynamics:

- Recognizes how subject writes character string
- Examines **how** subject performs writing AND features in written sample
- **Success relies on**: Pen pressure, stroke pattern, stroke length, points when pen lifted from paper
- **Speed usually NOT important factor**

Keystroke Patterns (Keystroke Dynamics):

- Analyzes how subject uses keyboard
- **Flight time**: How long between key presses
- **Dwell time**: How long key is pressed
- **Advantages**: Inexpensive, nonintrusive, often transparent to user (use and enrollment)
- **Disadvantages**: Subject to wild variances from simple behavior changes
- **Affected by**: Using one hand, being cold, standing vs. sitting, changing keyboards, hand/finger injury

Biometric Comparison Tools

Zephyr Chart: Rates various aspects/functions/features of different biometrics on single easy-to-read diagram

Biometric Rankings

Accuracy (Most to Least Accurate):

1. Palm scan
2. Hand geometry
3. Iris scan
4. Retina pattern
5. Fingerprint
6. Voice verification
7. Facial recognition
8. Signature dynamics
9. Keystroke dynamics

Acceptance (Most to Least Acceptable):

1. Iris scan
2. Keystroke dynamics
3. Signature dynamics
4. Voice verification
5. Facial recognition
6. Fingerprint
7. Palm scan
8. Hand geometry
9. Retina pattern

Appropriate Biometric Selection

When selecting biometric solution, consider:

- Which type most suitable for environment
- Effectiveness of biometric factor
- Acceptability of biometric factor
- Enrollment time
- Throughput rate
- Subject acceptance perceptions

For biometrics to be useful: Every subject and object must have stored biometric pattern; level of detail often results in false negatives and false positives

3. Tokens (Type 2)

Four Types of Token Devices:

Static Tokens

- **Examples**: Swipe card, smart card, floppy disk, USB RAM dongle, physical key
- Provide physical means to provide **identity**
- **Still require additional factor** for authentication (password or biometric)
- Most host cryptographic key (private key, digital signature, encrypted logon credentials)
- Cryptographic key **much stronger than password**: Pre-encrypted using strong encryption, significantly longer, resides only in token
- Most often used as **identification devices** rather than authentication factors

Synchronous Dynamic Password Tokens

- Generates passwords at **fixed time intervals**
- Requires **synchronizing clock** on authentication server with clock on token device
- Subject enters generated password + PIN/passphrase/password
- Generated password = identification; PIN/password = authentication

Asynchronous Dynamic Password Tokens

- Generates passwords based on **occurrence of an event**
- Requires subject press key on token AND on authentication server
- Action advances to next password value
- Generated password + subject's PIN/passphrase/password entered for authentication

Challenge-Response Tokens

- Generate passwords/responses based on instructions from authentication system

- Authentication system displays **challenge** (code or passphrase)
- Challenge entered into token device
- Token responds; response entered into system for authentication

Authentication Process: Involves workstation, token device, and authentication service working together

---

4. Tickets

Definition: Mechanism employing third-party entity to prove identification and provide authentication

Most Common System: **Kerberos**

- Developed under Project Athena at MIT
- Discussed in detail later in chapter

---

5. Single Sign-On (SSO)

Definition: Mechanism allowing subject to be authenticated only once and access resources without repeated authentication prompts

How It Works: After authentication, subject can roam network freely and access resources/services without further challenges

Advantages:

- Typically supports **stronger passwords** (subject must memorize only single password)
- Eases administration (reduces locations where account must be defined)

Primary Disadvantage:

- Once account compromised, malicious subject has **unrestricted access**
- **Maximum level of unauthorized access** gained simply through password disclosure

Implementation: Can be enabled through:

- Authentication systems
- Scripts providing logon credentials automatically when prompted

---

IV. Access Control Models & Techniques

Once subject is identified, authenticated, and accountability established, it must be **authorized** to access resources or perform actions.

Authorization occurs only AFTER identity verification through authentication

Access Controls: Systems provide authorization through use of access controls

Access Control Definition: Manage type and extent of access subjects have to objects

Two Primary Categories

1. **Discretionary**
2. **Nondiscretionary** (subdivided into: mandatory, role-based, task-based)

Each system has own security properties that distinguish it from others.

---

1. Discretionary Access Control (DAC)

Concept: Owner or creator of object controls and defines subject access to that object

Key Characteristic: Access control based on **discretion** (decision) of owner

Focus: Based on **identity** of subject (typically user account name)

Example: User creates spreadsheet file → user is owner → user can modify permissions to grant/deny access to other subjects

Implementation:

- Often implemented using **Access Control Lists (ACLs)** on objects
- Each ACL defines types of access granted or restricted to individual or grouped subjects

Access Control Structures

Access Control List (ACL):

- **Bound to an object**
- Defines which subjects can access object and what actions they can perform
- Example: File1 ACL shows User A (read), User B (read/write), User C (denied)

Capability Table:

- **Bound to a subject**
- Lists all objects subject can access and permissions for each
- Example: User A's capability table shows File1 (read), File2 (write), File3 (execute)

Self/Group/Public Controls:

- Common in Unix/Linux systems
- Permissions divided into three categories:
  - **Self (Owner)**: Permissions for file owner
  - **Group**: Permissions for group members
  - **Public (Others)**: Permissions for everyone else
- Each category can have read, write, execute permissions

---

2. Nondiscretionary Access Control

Concept: Rule-based system where set of rules, restrictions, or filters determines what can/cannot occur

Key Characteristics:

- Access **NOT** based on administrator or owner discretion
- **NOT** focused on user identity
- Access managed by **static set of rules** governing whole environment (centrally controlled)
- Opposite of discretionary (Non-A is opposite of A)

Appropriate For: Environments experiencing frequent changes to data permissions

Reason: Rule-based systems implement sweeping changes by changing central rules without manipulating every subject/object

Reality: Once rules established, they remain fairly static and unchanged throughout environment life

A. Rule-Based Access Control

Definition: Control based on specific profile created for each user

Common Example: **Firewall**

- Governed by set of rules/filters defined by administrator
- Users communicate across firewall because they initiated transactions allowed by defined rules
- Users accomplish this through client environments configured to do so (specific profiles)

Formalized Definition (RFC 2828 "Internet Security Glossary"): "A security policy based on global rules imposed for all users. These rules usually rely on comparison of the sensitivity of the resource being accessed and the possession of corresponding attributes of users, a group of users, or entities acting on behalf of users."

B. Mandatory Access Control (MAC)

Foundation: Relies upon use of **classification labels**

Key Components:

- Each classification label represents **security domain** or **realm of security**
- **Security domain**: Realm of common trust governed by specific security policy for that domain
- **Subjects**: Labeled by level of **clearance** (form of privilege)
- **Objects**: Labeled by level of **classification** or **sensitivity**

Military Classification Labels (Highest to Lowest Sensitivity):

- Top Secret
- Secret
- Confidential
- Sensitive But Unclassified (SBU)
- Unclassified

Corporate/Commercial Classification Labels:

- Confidential
- Proprietary
- Private
- Sensitive
- Public

Access Rules:

- Subjects can access objects with **same or lower** level of classification
- Subjects with higher clearance granted access to highly sensitive resources **only if work tasks require such access**

Security Comparison:

- Generally recognized as **more secure than DAC**
- **Less flexible and scalable** than DAC
- TCSEC evaluation criteria lists mandatory protection as **higher level of security** than discretionary protection

Label Requirement: For MAC to function, **every subject and object must have security label**

Label Meanings: Can refer to:

- Sensitivity
- Value to organization
- Need for confidentiality
- Classification
- Department
- Project

Security classifications:

- Indicate **hierarchy of sensitivity**
- Each level is **distinct**

Three Types of MAC Environments

1. Hierarchical Environments:

- Classification labels in **ordered structure**
- Low security → Medium security → High security
- Linear progression of access levels

2. Compartmentalized Environments:

- **No relationship** between one security domain and another
- To gain access to object: Subject must have **exact specific clearance** for that object's security domain
- Each compartment is isolated
- No automatic access based on equivalent level

3. Hybrid Environments:

- **Combines** hierarchical and compartmentalized concepts
- Each hierarchical level may contain **numerous subcompartments** isolated from rest of security domain
- Provides **more granular control** over access
- Becomes **increasingly difficult to manage** as size increases (number of classifications, objects, subjects)

C. Role-Based Access Control (RBAC)

Concept: Subject's ability to access object defined via **subject roles** (job descriptions) or **tasks** (work functions)

Key Characteristic: Access depends on **job description** (role/task) rather than **subject identity**

Example: Management position = greater access than temporary job position

Advantage: Useful in **volatile environments with frequent personnel changes**

Comparison with DAC Groups:

Similarities:

- Both serve as containers to collect users into manageable units

Differences:

DAC Groups	RBAC Roles
User can belong to multiple groups	User may have only single role (traditional); new trends allow multiple roles
Collect rights/permissions from each group	Only rights/permissions assigned to role
Individual accounts can be directly assigned rights/permissions	No additional individually assigned rights/permissions
Access based on owner discretion	Access not determined by owner discretion
Focus control on identity of user	Access derives from inherent responsibilities of assigned role
	Access focuses on assigned role, not identity of user

Identity vs. Role:

- Two different users with same assigned role will have **same access and privileges**

Corporate Attractiveness:

- Increasingly attractive to corporations with **high employee turnover rates**
- Roles/job descriptions often **hierarchical**

- Higher roles created by adding access and privileges to lower roles
- **MAC and DAC environments can be replaced by RBAC solutions**

D. Task-Based Access Control (TBAC)

Concept: Basically same as RBAC

Key Difference: Instead of single role, each user assigned **dozens of tasks**

Task Definition: All relate to assigned work tasks for person associated with user account

Access Control: Based on rules (work tasks); focuses on **controlling access by tasks assigned** rather than user identity

E. Lattice-Based Access Control

Definition: Defines **upper and lower bounds of access** for every relationship between subject and object

Mechanism:

- Subjects acquire **least upper bound** of access to labeled objects
- Subjects acquire **greatest lower bound** of access to labeled objects
- Based on assigned **lattice positions**

Example: Subject with lattice permissions can:

- Access resources **up to** Private
- Access resources **down to** Sensitive
- **Cannot access**: Confidential, Proprietary, or Public resources

Original Purpose: Developed to address **information flow**

Primary Concern: Confidentiality

Common Example: Mandatory Access Control is lattice-based

Visual Representation: Boundaries provided by lattice create clear upper/lower limits on access matrix

V. Access Control Methodologies & Implementation

Two Primary Methodologies

1. Centralized Access Control

Definition: All authorization verification performed by **single entity** within system

Examples: RADIUS, TACACS

Advantages:

- Can be managed by **small team or individual**
- **Lower administrative overhead** (all changes in single location)
- **Single change affects entire system**

Disadvantages:

- **Single point of failure**
- If system elements cannot access centralized system, subjects and objects cannot interact

2. Decentralized (Distributed) Access Control

Definition: Authorization verification performed by **various entities** located throughout system

Examples: Domains and trusts

Advantages:

- **No single point of failure**
- If access control point fails, other points may balance load until repair
- Objects/subjects not relying on failed point can continue to interact normally

Disadvantages:

- Often requires **several teams or multiple individuals**
- **Higher administrative overhead** (changes implemented in numerous locations)

- **Maintaining homogeneity becomes more difficult** as number of access control points increases
- Changes to individual access control point affect only aspects relying on that specific point

---

VI. Access Control Administration

Definition: Collection of tasks and duties assigned to administrator to manage user accounts, access, and accountability

Foundation: System security based on **effective administration** of access controls

Four Principles: Identification, Authentication, Authorization, Accountability

Three Main Administrative Responsibilities

1. **User account management**
2. **Activity tracking**
3. **Access rights and permissions management**

---

1. Account Administration

Account Lifecycle Management

A. Creating New Accounts (Enrollment)

Security Procedure Requirements:

- Must be protected through organizational security policy procedures
- **Not created** at whim of administrator or at anyone's request
- Must follow **stringent procedure** flowing from HR department hiring/promotion procedures

Proper Creation Process:

1. HR department makes **formal request** for user account for new employee
2. Request must include **classification or security level** to be assigned
3. New employee's **department manager** verifies security assignment
4. Organization's **security administrator** verifies security assignment
5. Only after verification should account be created

Consequence of Improper Process: Creates holes and oversights exploitable by malicious subjects

Similar Process: Should be followed for increasing or decreasing existing user account security level

New Employee Training:

- As part of hiring process, train on organization security policies/procedures
- Before hiring complete, employees must **sign agreement** committing to uphold security standards
- Many organizations craft document stating: Violating security policy is grounds for **dismissal and prosecution** under federal, state, local laws

Credentials Distribution:

- When passing user account ID and temporary password, review:
  - Password policy
  - Acceptable use restrictions

Enrollment Process:

- Creates new identity
- Establishes factors system needs for authentication
- **Critical**: Enrollment must be completed **fully and accurately**
- **Critical**: Identity of individual being enrolled must be **proved**

Identity Verification Methods:

- Photo ID

- Birth certificate
- Background check
- Credit check
- Security clearance verification
- FBI database search
- Calling references

All are valid forms depending on organization's needs

B. Account Maintenance

Frequency Factors:

- Organizations with **static organizational hierarchies** and **low employee turnover/promotion**: Less maintenance required
- Organizations with **flexible/dynamic organizational hierarchies** and **high employee turnover/promotion**: Significantly more maintenance required

Primary Activity: Most account maintenance deals with **altering rights and privileges**

Procedures: Similar to new account creation procedures should govern access changes throughout user account life

Consequences: Unauthorized increases or decreases in account access capabilities can result in **serious security repercussions**

Temporary Employees:

- Should have **specific expiration dates** programmed into user accounts
- Maintains control established at account creation without requiring ongoing administrative oversight

C. Account Termination

When Employees Leave:

- User accounts should be **disabled, deleted, or revoked**
- Whenever possible, task should be **automated and tied to HR department**

- General rule: When paychecks stop, person should no longer have logon capabilities

Automation: Maintains control without constant administrative attention

Account, Log, and Journal Monitoring

Purpose:

- Activity auditing, account tracking, and system monitoring are **important aspects** of access control management
- **Without these capabilities**: Impossible to hold subjects accountable

Mechanism:

- Through establishment of identification, authentication, and authorization
- Tracking activities of subjects (including object access frequency) offers **direct and specific accountability**

Components:

- **User accounts**
- **Event logs**
- **System journals**

Function: Help piece together state of affairs for server at any referenced point along timeline of operation

Event Logs and System Journals:

- Capture events, changes, messages, other data describing activities on system
- Commonly used to support conclusions about incidents warranting investigation
- Example: When attacker exploits vulnerable service, server documented aspects in event logs/system journals

---

2. Access Rights and Permissions Management

Core Principle: Not all subjects should be granted access to all objects with same functional capabilities

Security Policy Implementation: Assigning access to objects is important part of implementing organizational security policy

Key Concepts:

- Not all subjects should access all objects
- Not all subjects should have same functional capabilities on objects
- Few specific subjects should access only some objects
- Certain functions accessible only to few specific subjects

Example: Data entry department does not require explicit access to accounting department resources/information

Specific Access Example: Only managers in accounting access financial data; only supervisors create and maintain that data

continue from slide 101 because Claude died

Chapter 2: Attacks and Monitoring

Introduction Access control = hardware/software/policies that grant/restrict access, monitor/record attempts, identify users, determine authorization. Covered in Chapters 1 & 2 for CISSP exam.

Monitoring

- Programmatic means holding subjects accountable while authenticated; detects unauthorized/abnormal activities
- Necessary for: detecting malicious actions, intrusion attempts, system failures; reconstructing events; providing prosecution evidence; producing problem reports/analysis
- Challenge: sufficient logging creates massive data volumes where important details get lost
- Solution: data reduction using data mining tools
- For automation/real-time analysis: intrusion detection systems (IDS) required

- Accountability maintained by recording subject/object activities and core system functions
- Audit trails evaluate system health/performance; system crashes indicate faulty programs/corrupt drivers/intrusion attempts
- Event logs leading to crashes help discover failure reasons
- Log files provide step-by-step history recreation
- Supports multiple defense layers including real-time detection/deterrence of network attacks (internal/external origin)

Intrusion Detection Systems (IDS)

- Automates audit log inspection and real-time system event monitoring
- Primary use: detect intrusions; also detects system failures, rates performance
- Watches for confidentiality/integrity/availability violations
- Goal: perpetrator accountability + timely/accurate intrusion response
- Recognizes attacks from: external connections (Internet/partner networks), viruses, malicious code, trusted internal subjects attempting unauthorized activities, unauthorized access from trusted locations
- Classification: technical detective security control

IDS Capabilities:

- Actively watches suspicious activity, peruses audit logs, sends administrator alerts on specific events
- Locks down system files/capabilities
- Tracks slow/fast intrusion attempts
- Highlights vulnerabilities
- Identifies intrusion origination point
- Tracks perpetrator logical/physical location
- Terminates/interrupts attacks/intrusion attempts
- Reconfigures routers/firewalls to prevent attack repeats
- Alert methods: onscreen notification (most common), sound, email, pager, log file recording

IDS Response Types:

1. **Active response**: directly affects malicious activity/network traffic/host application
2. **Passive response**: doesn't affect malicious activity; records information + notifies administrator
3. **Hybrid response**: stops unwanted activity + records information + possibly notifies administrator

- IDS detects unauthorized/malicious activity from inside/outside trusted networks
- Limited capability to stop current attacks/prevent future attacks
- Typical responses: blocking ports, blocking source addresses, disabling communications over specific cable segments
- When discovering abnormal traffic (spoofed traffic) or security policy/filter/rule violations: records log detail, then drops/discards/deletes relevant packets

IDS as Security Component

- One component of well-formed security; complements firewalls
- Other necessary controls: physical restrictions, logical access controls
- Intrusion prevention requires: adequate system security maintenance (patches, security controls); responding to discovered intrusions by erecting barriers against future occurrences
- Responses range from: simple (updating software/reconfiguring access controls) to drastic (reconfiguring firewall, removing/replacing applications/services, redesigning entire network)

IDS vs IPS Architecture: [diagram showing difference]

Intrusion Response Protocol:

- First response: contain intrusion (prevents additional damage but may allow continued infestation of compromised systems)
- After containment: rebuild compromised systems from scratch

- Before reconnecting: double-check security policy compliance (ACLs, service configurations, user account settings)
- IDS type/classification defines scope of responsibility/functional role

Host-Based IDS:

- Watches questionable activity on single computer system
- Focused attention on single computer enables much greater event detail examination than network-based IDS
- Can pinpoint files/processes compromised/employed by malicious users for unauthorized activity
- Detects anomalies undetected by network-based IDS
- **Limitations:** cannot detect network-only attacks or attacks on other systems; attackers can discover/disable/manipulate IDS to hide tracks; difficulty detecting/tracking DoS attacks (especially bandwidth consumption); consumes monitored computer resources reducing system performance; limited by host OS/application auditing capabilities
- **Management:** more costly than network-based IDS; requires installation on each monitored server + administrative attention at each point; causes significant host performance degradation; easier for intruders to discover/disable

Network-Based IDS:

- Detects attacks/event anomalies through network packet capture/evaluation
- Single network-based IDS monitors large networks if installed on backbone (where majority traffic occurs)
- Some versions use remote agents collecting subnet data reporting to central management console
- Installed on single-purpose computers enabling: hardening against attack, reduced vulnerabilities, stealth mode operation
- **Stealth mode:** IDS invisible to network; intruders must know exact location/system identification to discover
- Little negative effect on overall network performance; doesn't adversely affect other computer performance (single-purpose system deployment)

- **Limitations:** cannot monitor encrypted traffic content; detects attack initiation/ongoing attempts but cannot provide information about attack success or which systems/accounts/files/applications affected; offers limited functionality for discovering attack sources via RARP/DNS lookups
- Performance issues if network traffic load high and unable to process packets efficiently/swiftly; cannot examine encrypted traffic contents

Network-Based IDS Architecture: [diagram]

IDS as Part of Multi-faceted Solution:

- Not single universal security solution; part of multi-faceted security solution
- Host-based IDS may miss details if host system overworked with insufficient IDS process execution time
- Network-based IDS suffers same problem with high network traffic loads

Knowledge-Based (Signature-Based) Detection:

- Uses signature database attempting to match all monitored events
- Event matches = IDS assumes attack occurring/occurred
- IDS vendor develops suspect chart by examining/inspecting numerous intrusions on various systems
- Results in description/signature of common attack methods/behaviors
- Functions like many antivirus applications
- **Primary drawback:** only effective against known attack methods; new attacks/slightly modified known attacks often unrecognized; lacks learning model (unable to recognize new attack patterns as they occur); only useful as signature file is correct/up-to-date
- Keeping signature file current = important aspect maintaining best performance

Behavior-Based Detection (Statistical/Anomaly/Heuristics-Based):

- Learns normal system activities/events through watching/learning
- After accumulating sufficient normal activity data: detects abnormal/possible malicious activities/events
- Can be labeled expert system/pseudo-artificial intelligence (can learn and make assumptions)
- Acts like human expert evaluating current events against known events
- More information about normal activities = more accurate anomaly detection
- **Primary drawback:** produces many false alarms; normal user/system activity patterns vary widely making definition of normal/acceptable activity difficult; more false alarms = less likely administrators heed warnings
- Over time becomes more efficient/accurate but learning process takes considerable time
- Using known behaviors, activity statistics, heuristic evaluation: detects unforeseen/new/unknown vulnerabilities, attacks, intrusion methods

Alarm Systems:

- Both detection methods employ alarm-signal systems
- When intrusion recognized/detected: alarm triggered
- Alarm system can: notify administrators (email/popup messages), execute scripts sending pager messages, record alert messages in log/audit files, generate violation reports detailing detected intrusions/discovered vulnerabilities

Detection Type Comparison:

Signature-based:

- Pattern matching (similar to antivirus software)
- Signatures must be continuously updated
- Cannot identify new attacks
- Two types:
  - Pattern matching: compares packets to signatures

    - Stateful matching: compares patterns to several activities at once

Anomaly-based:

- Behavioral-based system learning "normal" environment activities
- Can detect new attacks
- Also called behavior-/heuristic-based
- Three types:
    - Statistical anomaly-based: creates "normal" profile, compares activities to profile
    - Protocol anomaly-based: identifies protocols used outside common bounds
    - Traffic anomaly-based: identifies unusual network traffic activity

IDS-Related Tools: Deployed in concert to expand usefulness/capabilities, improve efficiency, reduce false positives:

1. Honey Pots:

- Individual computers/entire networks created as intruder snare
- Look/act like legitimate networks but 100% fake
- Honey pot access detected = most likely unauthorized intruder
- Deployed to keep intruder logged on performing malicious activities long enough for automated IDS to: detect intrusion, gather intruder information
- Longer honey pot retention = more administrator investigation time + potential intruder identification

2. Padded Cells:

- Similar to honey pot but different intrusion isolation approach
- When IDS detects intruder: automatically transferred to padded cell
- Padded cell has actual network look/layout but intruder cannot: perform malicious activities, access confidential data

- Simulated environment offers fake data retaining intruder interest
- Transfer performed without informing intruder of change
- Heavily monitored; administrators gather evidence for tracing/possible prosecution

3. Vulnerability Scanners:

- Test systems for known security vulnerabilities/weaknesses
- Generate reports indicating areas/aspects needing management to improve security
- Reports may recommend: applying patches, making configuration changes, making security setting changes
- Only useful as database is current; requires frequent vendor updates
- Used with IDS may reduce false positives + keep total intrusions/security violations minimal
- Quick/frequent vulnerability patching = more secure environment

4. Intrusion Prevention System (IPS):

- Extension of IDS concept
- Actively blocks unauthorized connection attempts/illicit traffic patterns as they occur
- Falls under same type (host-/network-based) and classification (behavior-/signature-based) as IDS
- Often deployed together for complete network coverage
- Many platforms capable of dissecting higher-level application protocols searching for malicious payloads

Penetration Testing

- Penetration = successful attack where intruder breaches environment perimeter
- Breach ranges from: reading few data bits to logging in as unrestricted privileges user
- Primary security goal: prevent penetrations
- **Penetration testing:** vigorous attempt breaking into protected network using any means necessary

- Common practice: hire external consultants (not privy to confidential security configuration/network design/internal secrets)
- Seeks finding all detectable weaknesses in existing security perimeter
- Note: undetected/presently unknowable threats exist in large-scale infrastructure (no penetration testing amount can directly discover/reveal)
- Once weakness discovered: countermeasures selected/deployed improving environment security
- **Key difference from actual attacking:** once vulnerability discovered, intrusion attempt ceases before vulnerability actually exploited/causes system damage
- Tools available: open source/commercial (Metasploit, Core IMPACT) attempting to exploit known vulnerabilities in systems/networks (used by good guys and bad guys)
- **Methods:** automated attack tools/suites or manual with common network utilities/scripting
- Automated tools range from: professional vulnerability scanners to wild underground attack tools from Internet
- Manual testing uses tools but places more onus on knowing how to perpetrate attacks
- **CRITICAL:** perform only with management consent/knowledge; unapproved security testing results in: productivity loss, emergency response team triggering, job loss, potential jail time

Methods of Attack

Common/Well-Known Attack Classes:

- Brute-force and dictionary attacks
- Denial-of-service attacks
- Spoofing
- Man-in-the-middle attacks
- Spamming
- Sniffers

All eventually attempted on networks; assess each as part of larger combination of risk potential/threat value rather than case-by-case severity assessment.

Simple one-stage attacks (brute-force/dictionary, spoofing, DoS): most common occurrences (easiest to mount, require only basic Internet accessibility)

Complex attacks (eavesdropping, sniffing, man-in-the-middle): involve intrusion component propelling attacker inside network perimeter

Brute-Force Attacks:

- Attempt discovering passwords for user accounts by systematically attempting every possible combination (letters/numbers/symbols)
- Modern computer speed + distributed computing ability = increasingly successful even against strong passwords
- Given enough time: all passwords discoverable
- Most passwords ≤14 characters discoverable within 7 days on fast system
- Longer password/greater algorithm key space number = more costly/time-consuming attack
- Increased possibilities = increased exhaustive attack cost
- Longer password = more secure against brute-force attacks

Dictionary Attacks:

- Attempt discovering passwords by attempting every possible password from predefined list of common/expected passwords
- Named such because possible password list so long = like using entire dictionary one word at a time
- Can be waged against: password database file or active logon prompt

Password Storage/Cracking:

- Passwords stored in account database file on secured systems
- Stored as hash values (not plain text) providing reasonable protection

- Using reverse hash matching: password attacker tool seeks possible passwords (brute-force/dictionary methods) with same hash value as account database file value
- Hash value match = tool has cracked password
- Combinations of brute-force/dictionary methodologies can be used (e.g., brute-force using dictionary list as guesswork source)

Protecting Passwords Requires:

- Controlling physical access to systems (password file)
- Controlling electronic access to password files (tightly control/monitor)
- Creating strong password policy
- Deploying two-factor authentication
- Using account lockout controls (prevents brute-force/dictionary attacks against logon prompts)
- Encrypting password files

Denial-of-Service (DoS) Attacks:

- Prevent system from processing/responding to legitimate traffic/resource requests
- Most common form: transmitting so many data packets server cannot process them all
- Other forms: exploiting known OS/service/application faults/vulnerabilities (often results in system crash/100% CPU utilization)
- Any attack rendering victim unable to perform normal activities = DoS attack
- Can result in: system crashes, system reboots, data corruption, service blockage
- Flooding-based DoS attacks = way of life on Internet

DoS Flood Attack Types:

1. Original/Simple DoS:

- Single attacking system flooding single victim with steady packet stream

- Packets could be: valid incomplete requests or malformed/fragmented packets consuming victim system attention
- Easy to terminate by blocking source IP address packets

2. Distributed Denial of Service (DDoS):

- Attacker compromises several systems using them as launching platforms against one/more victims
- Compromised systems = slaves/zombies
- Results in victims flooded with data from numerous sources

3. Distributed Reflective Denial of Service (DRDoS):

- More recent form
- Takes advantage of normal operation mechanisms of key Internet services (DNS, router update protocols)
- Functions by sending numerous update/session/control packets to various Internet service servers/routers with spoofed source address of intended victim
- Usually targets high-speed/high-volume Internet backbone trunk systems/routers
- Results in flood of: update packets, session acknowledgment responses, error messages sent to victim
- Can create so much traffic upstream systems adversely affected by sheer data volume focused on victim
- Called reflective attack because high-speed backbone systems reflect attack to victim
- Cannot be prevented (exploits normal system functions)
- Blocking packets from key Internet systems = effectively cutting victim from significant Internet section

Non-Malicious DoS:

- Not all DoS instances result from malicious attack
- Errors in coding OS/services/applications result in DoS conditions
- Examples: process failing to release CPU control, service consuming system resources disproportionate to service requests handled

- Most vendors quickly release patches correcting self-inflicted DoS conditions
- Important to stay informed

SYN Flood Attacks:

- Wages attack by breaking standard TCP/IP three-way handshake initiating communication sessions:
  1. Client sends SYN packet to server
  2. Server responds with SYN/ACK packet to client
  3. Client responds with ACK packet back to server
- Three-way handshake establishes communication session used for data transfer until terminated
- **SYN flood occurs:** numerous SYN packets sent to server but sender never replies to server's SYN/ACK packets with final ACK
- Transmitted SYN packets usually have spoofed source address (SYN/ACK response sent somewhere other than actual packet originator)
- Server waits for client's ACK packet (often several seconds) holding open session consuming system resources
- Significant number of open sessions (through SYN packet flood receipt) = results in DoS
- Server easily overtaxed by keeping never-finalized sessions open causing failure
- Failure ranges from: unable to respond to legitimate communication requests to frozen/crashed system
- **Countermeasures:**
  - Increasing server-supported connections number (usually requires additional hardware: memory, CPU speed; may not be possible for all OS/network services)
  - More useful: reduce timeout period waiting for final ACK packet (can result in failed sessions from slower link clients or intermittent Internet traffic hindrance)
  - Network-based IDS may offer protection against sustained attacks by noticing numerous SYN packets from one/few locations resulting in incomplete sessions; IDS could warn of attack or dynamically block flooding attempts

Smurf Attacks:

- Occurs when amplifying server/network used to flood victim with useless data
- Common attack: send message to subnet/network broadcast server so every node produces one/more response packets
- Attacker sends information request packets with victim's spoofed source address to amplification system (all response packets sent to victim)
- **Scenario:**
  1. Attacking machine sends ping request to one/more broadcast servers falsifying source IP address (provides target machine IP address)
  2. Broadcast server passes request to entire network
  3. All network machines send response to broadcast server
  4. Broadcast server redirects responses to target machine
- Ping exploits ICMP protocol testing network connections by sending packet/waiting for response
- **Countermeasures:** disabling directed broadcasts on all network border routers; configuring all systems to drop ICMP ECHO packets; IDS may detect but no prevention means other than blocking amplification network addresses (problematic because amplification network usually also victim)

Ping-of-Death Attack:

- Dates back to 1990s but uses interesting/devastating (at creation time) techniques subverting incoming IP data handling
- Specific defenses erected so attacks have very little success chance today
- Employs oversized ping packet
- Using special tools: attacker sends numerous oversized ping packets to victim
- Often when victimized system attempts processing packets: error occurs causing system freeze/crash/reboot
- More of buffer-overflow attack but considered DoS attack because often results in downed server

- **Countermeasures:** keeping up-to-date with OS/software patches; properly coding in-house applications preventing buffer overflows; avoiding running code with system-/root-level privileges; blocking ping packets at border routers/firewalls

WinNuke Attack:

- Specialized assault against Windows 95 systems
- Out-of-band TCP data sent to victim system causing OS freeze
- **Countermeasures:** updating Windows 95 with appropriate patch or changing to different OS

Stream Attack:

- Large number of packets sent to numerous victim system ports using random source/sequence numbers
- Victim system processing attempting to make sense of data results in DoS
- **Countermeasures:** patching system; using IDS for dynamic blocking

Teardrop Attack:

- Attacker exploits OS bug
- Bug exists in routines used to reassemble (resequence) fragmented packets
- Attacker sends numerous specially formatted fragmented packets to victim causing system freeze/crash
- **Countermeasures:** patching OS; deploying IDS for detection/dynamic blocking

Land Attack:

- Attacker sends numerous SYN packets to victim with SYN packets spoofed to use same source/destination IP address/port number as victim
- Causes system to think it sent TCP/IP session opening packet to itself causing system failure usually resulting in system freeze/crash/reboot
- **Countermeasures:** patching OS; deploying IDS for detection/dynamic blocking

Botnets:

- All older attack methods well documented today granting: zero stealth, minimal effectiveness against modern computing networks/operating systems
- More troubling trend emerged in recent years including botnets rise
- Coordinated networks of compromised machines used cohesively/scheduled manner to: attack, compromise, disrupt other end users/entire networks
- Widely employed to: distribute spam on behalf of third parties seeking paying customers through unwanted email, disseminate phishing lures parting unwary/naive recipients from cash
- Every botnet usually has one/more controlling computers (botnet controllers) providing cutouts between actual botnet operator (bot herder) and compromised machines
- Enables bot herders to: control larger computer numbers (many botnets exceed 100,000 compromised PCs; some instances exceed million machines reported 2007/2008), protect themselves from discovery even if botnets detected/disabled
- With hundreds of thousands to millions potential attacking machines in corrals: botnet ability to mount huge/devastating DoS attacks painfully obvious
- Recently used to slow down/deny access to global portals including Google, Yahoo, Microsoft

Spoofing Attacks:

- Art of pretending to be something other than what you are
- Consists of replacing valid source/destination IP address/node numbers with false ones
- Involved in most attacks granting attackers ability to hide identity through misdirection
- Employed when: intruder uses stolen username/password to gain entry, attacker changes malicious packet source address, attacker assumes client identity fooling server into transmitting controlled data

Two Specific Spoofing Attack Types:

1. Impersonation:

- More active attack
- Requires capturing authentication traffic + replaying traffic to gain system access

2. Masquerading:

- More passive attack
- Attacker uses previously stolen account credentials to log on to secured system
- Ultimately same attack: someone gains secured system access by pretending to be someone else
- Often results in unauthorized person gaining system access through compromised valid user account

Man-in-the-Middle Attacks:

- Occurs when malicious user gains position between two endpoints of ongoing communication
- **Two types:**
  1. **Copying/sniffing traffic between two parties:** basically sniffer attack
  2. **Positioning in communication line as store-and-forward/proxy mechanism:** attacker functions as receiver for client-transmitted data and transmitter for server-sent data; attacker invisible to both communication link ends; able to alter content/flow of traffic; through this attack: attacker collects logon credentials/sensitive data and changes message content exchanged between endpoints
- To perform: attacker must often alter routing information/DNS values, steal IP addresses, defraud ARP lookups to impersonate server from client perspective and impersonate client from server perspective

Hijack Attack:

- Offshoot of man-in-the-middle attack
- Malicious user positioned between client/server then interrupts session and takes it over

- Often malicious user impersonates client to extract server data
- Server unaware any communication partner change occurred
- Client aware server communications ceased but no indication why communications terminated available

Replay Attack (Playback Attack):

- Similar to hijacking
- Malicious user records client/server traffic; then client-to-server packets played back/retransmitted to server with slight time stamp/source IP address variations (spoofing)
- In some cases: allows malicious user to restart old server communication link
- Once communication session reopened: malicious user attempts obtaining data/additional access
- Captured traffic often authentication traffic (typically includes logon credentials: username/password) but could also be service access traffic/message control traffic
- **Prevention:** employing complex sequencing rules/time stamps preventing retransmitted packets from being accepted as valid

Man-in-the-Middle Attack Countermeasures:

- Require improvement in: session establishment, identification, authentication processes
- Some attacks thwarted through OS/software patching
- IDS cannot usually detect man-in-the-middle/hijack attacks but can often detect abnormal activities occurring via "secured" communication links
- Operating systems/many IDSs can often detect/block replay attacks

Sniffer Attacks (Snooping Attacks):

- Any activity resulting in malicious user obtaining network/network traffic information
- Sniffer often = packet-capturing program duplicating network medium traveling packet contents into file
- Often focus on initial client/server connections to obtain logon credentials (usernames/passwords), secret keys, etc.

- When performed properly: invisible to all other network entities; often precede spoofing/hijack attacks
- Replay attack = type of sniffer attack
- **Countermeasures:** improving physical access control; actively monitoring for sniffing signatures (packet delay, additional routing hops, lost packets - some IDSs can perform); using encrypted traffic over internal/external network connections

Spamming Attacks:

- Spam = unwanted email/newsgroup/discussion forum messages
- Ranges from: innocuous vendor advertisement to malignant floods of unrequested messages with viruses/Trojan horses attached
- Usually not security threat but rather DoS attack type
- As spam level increases: locating/accessing legitimate messages difficult
- Beyond nuisance value: spam consumes significant Internet resources portion (bandwidth/CPU processing) resulting in overall slower Internet performance/lower bandwidth availability for everyone
- Spamming attacks = directed floods of unwanted messages to victim's email inbox/other messaging system
- Causes DoS issues by filling storage space/preventing legitimate message delivery
- Extreme cases: causes system freezes/crashes, interrupts other user activity on same subnet/ISP
- **Countermeasures:** using email filters, email proxies, IDSs to detect/track/terminate spam flood attempts

Crackers, Hackers, and Attackers:

- **Crackers:** malicious users intent on waging attacks against person/system; may be motivated by greed/power/recognition; actions result in: stolen property (data/ideas), disabled systems, compromised security, negative public opinion, market share loss, reduced profitability, lost productivity
- **Hackers:** technology enthusiasts with no malicious intent (term commonly confused with crackers)
- Many authors/media use "hacker" when discussing cracker issues

- Book uses "attacker" for malicious intruders to avoid confusion
- Thwarting attacker attempts requires: vigilant effort keeping systems patched/properly configured; IDSs/honey pot systems often offer means to detect/gather prosecution evidence once controlled perimeter breached

Access Control Compensations:

- Access control regulates/specifies which objects subject can access and what access type allowed/denied
- Numerous attacks designed to bypass/subvert access control
- Beyond specific countermeasures for each attack: certain measures help compensate for access control violations
- Compensation measure = not direct problem prevention but means to design environment resiliency providing support for quick recovery/response

Compensation Measures:

1. Backups:

- Best means to compensate against access control violations
- With reliable backups/data restoration mechanism: any corruption/file-based asset loss can be repaired/corrected/restored promptly
- RAID technology provides fault tolerance allowing quick recovery in device failure event/severe access violation

2. Avoiding Single Points of Failure:

- Deploying fault-tolerant systems helps ensure loss of use/control over single system/device/asset doesn't directly lead to entire network environment compromise/failure
- Fault tolerance countermeasures designed to combat design reliability threats
- Having: backup communication routes, mirrored servers, clustered systems, failover systems = provides instant automatic/quick manual recovery in access control violation event

3. Business Continuity Planning:

- Should include procedures dealing with access control violations threatening mission-critical process stability

4. Insurance Coverage:

- Should include asset categories requiring compensation in severe access control violation event

CHAPTER 4: DATA AND APPLICATION SECURITY (PARTS 1 & 2)

TYPES OF APPLICATIONS (PART 1)

Agents:

- Small standalone programs part of larger applications
- Carry out specific functions: remote status collection, remote system management
- Examples:
  - **Anti-virus:** workstation/server agent in enterprise environment with central management console
  - **Patch management:** agent on each server periodically queries OS for software patch existence; installs patches when commanded from central patch management server
  - **Configuration management:** central server tracks/manages OS configuration of each server/workstation by communicating to agents on managed systems; agents collect configuration information passing it to central servers; agents perform configuration changes upon command

Applets:

- Software program running within context of another program
- Unable to run on its own; performs narrow function
- Most common use: within Web browsers
- Examples: media players (Flash, Shockwave players), content viewers (Adobe Reader), Java applets running in Web browser windows

Client-Server Applications:

- Software components not centralized; present in two places: clients and servers
- Clients/servers usually communicate via networks
- **Client characteristics:** software part used by humans; primarily contains user interface logic displaying instructions/data, accepting input data from keyboard/other device, accepting instructions/directives from users
- **Server characteristics:** typical client-server applications have server component performing database updates and communicating with clients; server components typically lack user interface logic but instead run as daemons/services

Distributed Applications:

- Software components running on several separate systems in wide variety of architecture: two-tier, three-tier, multi-tier
- Usually designed to physically/logically separate different application functions
- Many possible separation reasons: scalability, performance, geographical, security

Web Applications: Provide several significant advances over client-server applications:

- **Thinner clients:** browser becomes client software working with all enterprise's web applications
- **Better network performance:** more business logic resides on server; only display logic resides on workstation significantly reducing network demands; enables more users without network meltdowns
- **Lower cost of ownership:** organization only needs reasonably current web browser version on workstations; eliminates administrative overhead maintaining client software component versions for all organization's client-server applications
- **More terminal types supported:** users no longer locked into hardware/OS platform; can access applications using variety of terminal types (Windows, UNIX, Apple workstations, mobile devices: smartphones, PDAs)

- **Any user can access application:** because web application requires only browser as client, any user anywhere in world can potentially access application

APPLICATION MODELS AND TECHNOLOGIES

Computer systems/application programming languages generally built upon models giving language form/structure. Four most popular models:

1. Control Flow Languages:

- Earliest computer languages; sequential in nature (executed statements one after another)
- Most languages used variation of "if-then" construct + "goto" construct to alter instruction sequence
- **Disadvantage:** difficulty verifying program integrity; excessive "goto" use turned linear logic into "spaghetti" code (difficult to analyze/understand)
- "goto" statement demonized; structured languages won favor

2. Structured Languages:

- Programming languages with procedural structure developed to overcome control flow application deficiencies with "goto" statements
- Used subroutines/functions; relied less on goto (some structured languages lack goto entirely)
- Tend to be structured in code "blocks" bracketed by keywords: if…fi, BEGIN…END, {…}, if…then…else…endif, etc.
- Logic flow tends to be hierarchical rather than linear making analysis/verification somewhat easier

3. Object Oriented (OO) Programming: Completely different approach than structured languages (BASIC, C). Particular vocabulary describing how components are named/assembled into programs:

- **Class:** defines object characteristics including attributes, properties, fields, plus methods it can perform
- **Object:** particular instance of a class (example: class superhero defines all superheroes listing characteristics;

object Superman = one particular superhero; object Superman = instance of class superhero)

- **Method:** defines abilities object can perform; may contain instructions, input variables, output variables; similar to function/subroutine in structured programming; consists of instructions/calculations; communicates using message passing (example: fly() = one of Superman's methods)
- **Encapsulation:** refers to implementation details in method that are concealed (example: Superman's fly() method code contains several other methods like propulsion(), steering() that other objects don't need concern about)
- **Inheritance:** refers to subclass characteristics inheriting attributes from parent classes; in turn, subclasses can introduce own attributes passed to their subclasses
- **Polymorphism:** characteristic allowing objects of different types to respond to method calls differently depending upon type (example: compute-tax() method call results in different behavior depending upon country/type where transaction takes place - different tax rates, different taxable goods, some people taxed at different rates)

4. Knowledge-Based Applications:

Knowledge-based systems:

- Applications used to make predictions/decisions based upon input data
- Include feedback mechanisms enabling them to learn and refine guidance improving accuracy over time
- Objective: system ability to possess some human reasoning qualities

Neural Networks:

- Named because modeled after biological reasoning processes humans possess
- Consists of interconnected artificial neurons storing information pieces about particular problem
- Given many cases of situations/outcomes; more events given = more accurately predicts future outcomes

Expert Systems:

- Accumulate knowledge on particular subject including conditions/outcomes
- More samples expert system obtains = greater ability to predict future outcomes

THREATS IN THE SOFTWARE ENVIRONMENT (PART 2)

1. BUFFER OVERFLOW:

- Software applications usually function by soliciting/accepting input from user (or another application) through interface
- Attacker can attempt disrupting software application function by providing more data than application designed to handle
- Buffer overflow attack occurs when someone attempts disrupting program operation in this manner

Types of Buffer Overflow Attacks:

- **Stack Buffer Overflow:** program writes more data to stack-located buffer than allocated; causes other stack data corruption resulting in program malfunction
- **NOP Sled Attack:** specific stack overflow attack where attacker overflows stack with harmless NOP (no-op) instructions; point: improve chances attacker will find attack point; by flooding stack with lots of NOPs, program encounters and "slides down" NOPs until reaching pointer attacker placed in buffer
- **Heap Overflow:** heap = dynamically allocated memory space created by program for variable storage; usually heap overflow attack results in corruption of other variables already on heap
- **Jump-to-Register Attack:** return pointer overwritten with value causing program to jump to known pointer stored in register pointing to input buffer

Buffer Overflow Countermeasures: Several tactical/strategic countermeasures available to reduce/eliminate buffer overflow attack risk. Used to either remove buffer overflow capabilities or detect/block buffer overflow activity:

- **Choose safe language:** C/C++ don't automatically check input buffer lengths or perform other boundary checking; Java, .NET, many other languages have built-in boundary checking that (in most cases) prevents buffer overflows
- **Use of safe libraries**
- **Executable space protection:** feature of some operating systems forces programs to abort if attempting to execute code in stack/heap; some CPUs support executable space protection in hardware
- **Stack smashing protection:** refers to techniques used to detect stack changes
- **Application firewalls**

2. MALICIOUS SOFTWARE (MALWARE):

Malicious software (malicious code) = class of software in many forms performing variety of damaging actions.

Purposes of malware:

- **Propagation:** sometimes ability for malware to propagate (spread from system to system) = only purpose for particular malware programs
- **Damage and destruction of information:** malware can alter/delete files on target systems
- **Steal information:** malware can locate/steal valuable information: e-mail addresses, userids/passwords, bank account numbers, credit card numbers
- **Usage monitoring:** malware can implant means to record subsequent communications, keystrokes/mouse clicks, sending data back to malware's owner-operator
- **Denial of Service:** malware can consume all available target system resources rendering it essentially useless for intended use
- **Remote control:** malware can implant bot onto target system allowing attacker to remotely control system; large bot collections = bot armies; people who build/control bot armies = bot herders/botnet operators

Types of Malicious Software:

A. Viruses:

- Computer code fragments attaching themselves to legitimate program file on computer
- Virus can only run when legitimate program run
- Viruses generally require human intervention to propagate (user must run program to make virus spread)
- Viruses used to propagate through file sharing

Several virus types:

- **Master Boot Record (MBR) viruses:** viruses attach themselves to master boot record
- **File infector viruses:** viruses attaching themselves to executable programs (.EXE, .COM files)
- **Macro viruses**

Virus methods to avoid anti-virus detection:

- **Multipartite viruses:** use more than one means for propagating from one system to another (example: Ghostball infected both executable .COM program files and boot sectors)
- **Stealth viruses:** uses some means to hide itself from operating system detection
- **Polymorphic viruses:** virus creators introduced polymorphic viruses that change themselves as they move from system to system to avoid detection; however, anti-virus company engineers able to solve polymorphic virus puzzle and create signature for them
- **Encrypted viruses:** viruses encrypt most of their code using different key on each infected system making most of virus body different on each detected system; however, virus part (decryption code) must remain same; this portion = what antivirus software must identify to stop virus

B. Worms: Generally speaking, worms = like viruses but usually require little human intervention to spread; instead have own built-in propagation means.

Two common worm types:

- **Mass mailing worms:** propagate via e-mail; generally when mass-mailing worm arrives in user's inbox, worm activated when recipient opens message; worm's malicious code could reside within message HTML code or in attached file
- **Port scanning worms:** able to propagate with no human intervention at all; scans network for other potentially vulnerable systems and attempts spreading to neighboring systems; if able to infect new system, installs itself and begins scanning looking for new victims

C. Trojan Horses: Like ancient Greek legend, computer-based Trojan horse = lie:

- Claims to be one thing but instead something else (something with more malicious intent)
- User running Trojan horse program may/may not see some visual resemblance of what program claims to be
- However, Trojan horse also performing some additional (probably malicious) action: corrupting/destroying files, stealing data, sending e-mails to your friends

D. Rootkits: One of newest malware forms; malware programs designed to avoid detection by being absolutely invisible to operating system:

- Rootkits achieve this by altering OS itself so presence nearly impossible to detect

Methods used by rootkits to avoid detection:

- **Process hiding:** rootkits can hide own process(es) from users by altering tools used to list processes on system
- **File hiding:** rootkits can hide files as way of avoiding detection
- **Registry hiding:** rootkits can hide registry entries attempting to function without detection
- **Running underneath OS:** rootkits can hide from OS by running underneath/beside it

E. Bots: Short for "robots"; bots sometimes = part of malicious payload found in malware:

- Enable "bot herder" (bot program owner) to remotely control infected computer for variety of purposes:
  - **Relaying spam:** technique spam blockers use to block spam by blocking all e-mail from specific IP addresses
  - **Hosting phishing sites**
  - **Denial of Service attacks:** bot herders can launch DoS attacks from bot-controlled systems by instructing those systems to launch thousands of network messages per second to target system; bot herder can launch distributed DoS (DDoS) attack by directing hundreds, thousands, tens of thousands of bot-systems to attack same target simultaneously

F. Spam: Unwanted e-mail; accounts for well over ninety percent of all Internet e-mail; more than that = unsolicited "junk mail" taking many forms:

- **Unsolicited commercial e-mail (UCE):** although UCE also used to market even legitimate goods/services, users often frown on this advertising type and frown at companies advertising this way
- **Phishing:** two-part attacks consisting of legitimate-looking e-mail messages from large/well-known organizations (often financial institutions) using some means to trick user into visiting web site; this web site resembles legitimate site asking for login/other credentials/information (credit card numbers, social insurance numbers, other information used to defraud user); most common phishing scams purport coming from banks asking users to log in and confirm account numbers/credit card numbers
- **Malware:** spam often used to directly deliver malware to users' computers; also often used to lure people to web sites containing malicious code (viruses, worms, bots)

G. Pharming: In pharming attack, attacker directs all traffic destined for particular web site towards imposter web site; attack diverts traffic by "poisoning" organization's DNS servers or by changing hosts file on individual users' systems

H. Spyware and Adware: Encompass wide variety of means developed to track users' Internet usage pattern behavior; while not strictly malicious, many find techniques/motives used by spyware/ad-ware suspicious and privacy invasion.

Spyware and adware forms:

- **Tracking cookies:** many web site operators track users' individual visits to web sites through tracking cookies that may accompany banner ads
- **Web beacons (web bugs):** tiny 1×1 pixel images embedded in web pages as means for tracking users' Internet usage; alternative to cookies; far more difficult to detect/block
- **Browser helper objects (BHOs):** sometimes take form of helpful toolbars but at other times completely invisible and "stealthy"; BHOs can be used to track users' Web browser use
- **Key loggers:** actually records user's keystrokes (often mouse movements/clicks) and transmits that data back to central location

MALICIOUS SOFTWARE COUNTERMEASURES:

Several measures needed to block malware ability to enter/run on system:

Anti-virus:

- Anti-virus programs run on system employing various means to detect possible malware entry and have ability to block its entry
- Anti-virus software can also remove malware if already present on system
- Anti-virus programs found in many organization places as part of defense in depth to prevent unwanted malware consequences
- Places where anti-virus software found: end user workstations, e-mail servers, file servers, web proxy servers, security appliances (all-in-one security appliances performing several functions: firewall, web content filter, spam filter, anti-virus)

Anti-rootkit Software: Uses techniques to find hidden processes, hidden registry entries, unexpected kernel hooks, hidden files to find rootkits that may be present on system; uses various means to find these hidden system objects

Anti-spyware Software: Software to block spyware/adware similar to anti-virus software: monitors incoming files examining them against signature collection; blocks files matching known signatures; like anti-virus software, anti-spyware can scan hard drive to identify spyware, adware, other unwanted programs; removes them as user-directed

Anti-spam Software: Spam blockers effectively eliminate most spam coming to organization; block majority of unwanted e-mail carrying malware, phishing scams, fraudulent advertising, porn.

Four common spam blocking architectures:

- **Client-based**
- **E-mail server-based**
- **Appliance-based**
- **Spam blocking service:** incoming e-mail delivered to off-site spam blocking service provider that filters out spam and delivers only legitimate e-mail to corporate e-mail servers

Firewalls: Time-tested and still-preferred means for blocking unwanted network traffic from crossing network boundary; typically used as perimeter devices protecting organizations from unwanted traffic originating from Internet

Decreased Privilege Levels:

- When malware successfully breaks into system and is executed by user, malware usually executing with same privilege level as user
- Side benefit of reducing user privileges to end user level = decreased tech support call number to repair uh-oh's when often-inexperienced end users muddle up operating system configurations

Penetration Testing: Rather than simply relying upon security configuration settings, organization should also test settings by using tools to simulate hacker's attempt to find system weaknesses; such tests = penetration tests (often known as "pen tests"); penetration testing object = discover/fix vulnerabilities before hacker able to discover/exploit them

Hardening: Server operating systems very complex and often pre-configured for wide variety of tasks; often means many available programs/features activated by default; result = server with necessary feature(s) activated plus many additional unnecessary features also activated and ready to accept input from any friendly/unfriendly party

3. INPUT ATTACKS:

Applications/tools often request user input; common attack method = provide data causing unexpected application behavior.

Input attacks (sometimes called malformed input attacks/injection attacks) designed to exploit application weaknesses by causing unexpected behavior:

- **Elevation of privileges:** attacker inputs specially coded data attempting to cause malfunction resulting in attacker having higher access/privilege level in application
- **Execution of arbitrary code:** attacker may wish to run specific commands on target system
- **Malfunction:** attacker may wish to cause application to malfunction and be in disabled state for legitimate users
- **Abort:** attacker may wish to cause application to completely abort thus being unavailable for any legitimate use

Types of Input Attacks:

- **Buffer overflow**
- **Integer overflow:** attack where attacker attempts causing application to perform integer operation creating numeric value larger than can be represented in available storage
- **SQL injection:** attacker inserts specially coded/delimited SQL statements into input field hoping injected SQL will be executed on back end

- **Script injection:** similar to SQL injection; attacker inserts script language into input field hoping scripting language will be executed
- **Cross-site scripting (XSS):** attack where attacker can inject malicious script into HTML content to steal session cookies and other sensitive information
- **Cross-site request forgery (XSRF):** attack where malicious HTML inserted into Web page/e-mail that when clicked causes action to occur on unrelated site where user may have active session

Input Attack Countermeasures:

- **Effective input field filtering:** input fields should be filtered to remove all characters that might be part of input injection
- **Application firewall:** application firewalls examine packet contents and block packets containing input attack code and other unwanted data
- **Application vulnerability scanning:** organizations developing own applications for online use should scan those applications for input attack vulnerabilities to identify vulnerabilities prior to being discovered/exploited by outsiders
- **Developer training:** software developers should be trained in secure application development techniques

4. LOGIC BOMB:

Logic bombs (sometimes known as time bombs) = instructions deliberately placed in application code that perform some hostile action when predetermined condition met.

- Typically logic bomb consists of code performing some damaging action on date in distant future
- Most often, developer plants logic bomb in application if he believes he will be terminated from employment
- Logic bomb will "go off" at some later date; terminated programmer feels he got his just revenge

Logic Bomb Countermeasures: Logic bombs/back doors very similar: both involve unwanted code in application. Countermeasures for logic bombs same as for back doors: code reviews, source code control, source code scanning, third party assessments

5. OBJECT REUSE:

Many system resources shared in multiprocessing systems including: memory, databases, file systems, paging space; when one process utilizes resource, process may write some information to resource temporarily

Object Reuse Countermeasures:

- **Application isolation:** applications should be isolated to individual systems; this way applications less likely to encounter residual information left by other applications
- **Server virtualization:** often not feasible to isolate applications to one-per-machine; however, virtualization technology may make it more cost-effective to isolate applications by running them on virtual machines
- **Developer training:** software developers can be shown how to write secure software that doesn't leave residual code usable by other processes

6. MOBILE CODE:

Also known as executable code, active content, downloadable content; mobile code can be downloaded/transferred from one system for execution on another system.

Examples of mobile code:

- **Active website content:** includes ActiveX, Java, JavaScript, Flash, Shockwave, etc.; this content originates on Web server and executes on user's workstation; depending upon technology associated with downloaded content, this mobile code may have restricted access to end user's system or may have full control over it
- **Downloaded software:** includes software of every kind from legitimate (and not-so-legitimate) sites; some software may be purely benign but others can be Trojan horse programs and worse; some = outright malware with/without disguise

Mobile Code Countermeasures:

- **Anti-malware:** includes anti-virus, anti-spyware, etc.; these protective programs should be in place, properly configured, up-to-date
- **Reduced user privileges:** end users should not be permitted to install/execute mobile code on workstations except in explicitly permitted situations (company produced mobile code)
- **Mobile code access controls:** access controls should be in place to prevent unauthorized persons from downloading any mobile code they're not permitted to access/use
- **Secure workstation configuration:** workstations should be configured to restrict mobile code except in cases where specific mobile code permitted; may involve centralized workstation configuration that cannot be defeated/circumvented by end users

7. SOCIAL ENGINEERING:

Social engineering attack = attack on organization personnel; usually purpose = gain secrets from individuals that can later be used to gain unauthorized system access.

- Social engineer uses technique known as **pretexting** in effort to pretend they are someone else
- Social engineering owes success to basic human nature: people willing to help others in need and "be the hero"; social engineers prey on this weakness in feigned calls for assistance

Social Engineering Countermeasures: Best countermeasure against social engineering = education: people in organization (particularly those with administrative privileges: system administrators, network administrators, database administrators, etc.) need to be educated on proper procedures for providing company sensitive information to others

8. BACK DOOR:

Back door = mechanism deliberately planted in system by application developer that allows developer/other person to circumvent security.

Back doors may be present in application for several reasons:

- **To facilitate testing during application development:** for instance, back doors can be activated by entering specific values causing program to enter interactive debug mode
- **To facilitate production access:** for example, back door created so developer can access application while in production; this would be considered inappropriate back door use since developers should never have access to production application/production data
- **To facilitate break-in:** sometimes back doors inserted into application to permit unauthorized party to access application functions/data party should not have access to; this clearly inappropriate; this use resembles logic bomb

Back Door Countermeasures: Back doors can be difficult to find, particularly if inserted for disreputable purposes; routine functional testing/QA testing may not reveal back doors whatever their purpose. Instead, other means required to find them (same as Logic Bomb):

- **Code reviews:** when one developer makes changes to software application, one/more other developers should examine software to identify/approve all changes; should prevent both "legitimate" back doors and illegitimate ones
- **Source code control:** formal source code management system should be used that will identify/record all changes made to code
- **Source code scanning:** tools used to scan static source code for security vulnerabilities should be able to find back doors (or at least flag unusual logic associated with back door)
- **Third-party code reviews and assessments:** third-party organization more motivated to find software anomalies than its own developers