

TEL2813/IS2820 Security Management

Protection Mechanisms
Lecture 9
Feb 24, 2005



Introduction (Continued)

- Some of the most powerful and widely used technical security mechanisms include:
 - Access controls
 - Firewalls
 - Dial-up protection
 - Intrusion detection systems
 - Vulnerability
 - Auditing Systems

Sphere of Security

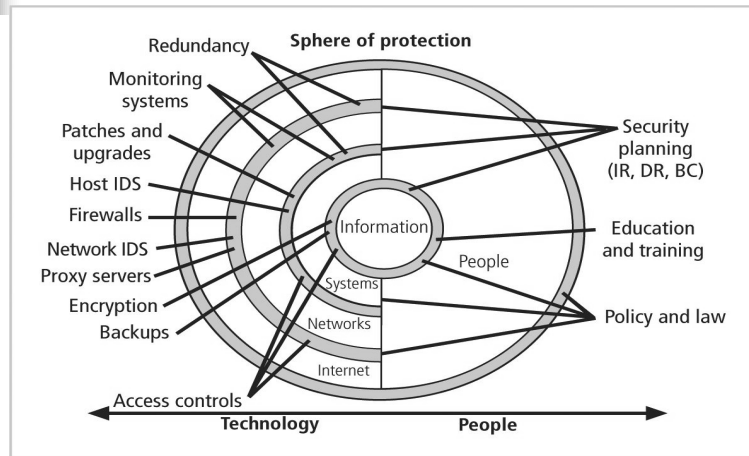


FIGURE 9-1 Sphere of Security

Access Control Devices

- Access control encompasses two processes:
 - Confirming identity of entity accessing a logical or physical area (authentication)
 - Determining which actions that entity can perform in that physical or logical area (authorization)
- A successful access control approach (for both physical access or logical access always consists of
 - authentication and
 - authorization



Authentication Mechanisms

- Mechanism types:
 - Something you know
 - Something you have
 - Something you are
 - Something you produce
- Strong authentication uses at least two different authentication mechanism types
 - Two factor authentication
 - Have + Know



Something You Know

- Authentication mechanism based on the user's identity
 - password, passphrase, or other unique code
 - A password is a private word or combination of characters that only the user should know
 - A passphrase is a plain-language phrase, typically longer than a password, from which a virtual password is derived
- A good rule of thumb is to require that passwords be at least eight characters long and contain at least one number and one special character
- Attack against password
 - Dictionary, brute force, man-in-the-middle, social engineering; keyboard attack

Password Power (1)

Table 9-1 Password Power

Case-Insensitive Passwords

Number of characters	Odds of cracking: 1 in	Estimated time to crack
1	68	0.000009 second
2	4624	0.0006 second
3	314,432	0.04 second
4	21,381,376	2.7 seconds
5	1,453,933,568	3 minutes, 2 seconds
6	98,867,482,624	3 hours, 26 minutes
7	6,722,988,818,432	9 days, 17 hours, 26 minutes
8	457,163,239,653,376	1 year, 10 months, 1 day
9	31,087,100,296,429,600	124 years, 11 months, 5 days
10	2,113,922,820,157,210,000	8495 years, 4 months, 17 days

Password Power (2)

Table 9-1 Password Power (continued)

Case-Sensitive Passwords

Number of characters	Odds of cracking: 1 in	Estimated time to crack
1	94	0.00001 second
2	8836	0.011 second
3	830,584	0.1 second
4	78,074,896	9.8 seconds
5	7,339,040,224	15 minutes, 17 seconds
6	689,869,781,056	23 hours, 57 minutes, 14 seconds
7	64,847,759,419,264	3 months, 3 days, 19 hours
8	6,095,689,385,410,820	24 years, 6 months
9	572,994,802,228,617,000	2302 years, 8 months, 9 days
10	53,861,511,409,490,000,000	216,457 years, 4 months

Something You Have

- Authentication mechanism based on what user has
 - a card, key, or token
 - dumb card (such as an ATM cards) with magnetic stripes
 - smart card containing a processor
- Cryptographic token, a processor in a card that has a display
- Tokens may be either
 - synchronous or
 - Synchronized with the server
 - Asynchronous
 - Challenge response



Source: RSA Security

FIGURE 9-3 Access Control Tokens

Something You Are

- Biometric
 - something inherent in the user
 - Fingerprints, palm scans, hand geometry/topology, facial recognition, retina scan, iris scan
- Most of the technologies that scan human characteristics convert these images to obtain some form of minutiae —
 - unique points of reference that are digitized and stored in an encrypted format



Something You Do

- This type of authentication makes use of something the user performs or produces
 - signature recognition and
 - voice recognition (voice phrase)
 - Key stroke pattern
 - Timing for known sequence of keystrokes



Authorization

- Authorization for each authenticated user
 - System performs authentication process to verify specific entity
 - Grants access to resources for only that entity
- Authorization for members of a group
 - System matches authenticated entities to a list of group memberships
 - Grants access to resources based on group's access rights
- Authorization across multiple systems
 - Central authentication and authorization system verifies entity identity
 - Grants a set of credentials to verified entity

Evaluating Biometrics

- False reject rate:
 - Percentage of authorized users who are denied access (Type I Error)
- False accept rate:
 - Percentage of unauthorized users who are allowed access (Type II Error)
- Crossover error rate:
 - Point at which the number of false rejections equals the false acceptances

Orders of Effectiveness and Acceptance

Table 9-2 Orders of Effectiveness and Acceptance

Effectiveness of Biometric Authentication Systems Ranking from Most Secure to Least Secure	Acceptance of Biometric Authentication Systems Ranking from Most Accepted to Least Accepted
■ Retina pattern recognition	■ Keystroke pattern recognition
■ Fingerprint recognition	■ Signature recognition
■ Handprint recognition	■ Voice pattern recognition
■ Voice pattern recognition	■ Handprint recognition
■ Keystroke pattern recognition	■ Fingerprint recognition
■ Signature recognition	■ Retina pattern recognition



Managing Access Controls

- To appropriately manage access controls, an organization must have a formal access control policy in place
 - Determines how access rights are granted to entities and groups
 - Must include provisions for periodically reviewing all access rights, granting access rights to new employees, changing access rights when job roles change, and revoking access rights as appropriate
- All those access control models !!!
 - ACM, SPM, BLP, Biba, Lipner, Clark-Wilson, RBAC



Perimeter Defense

- Organization system consists of a network of many host machines –
 - the system is as secure as the weakest link
- Use perimeter defense
 - Define a border and use gatekeeper (firewall)
- If host machines are scattered and need to use public network, use encryption
 - Virtual Private Networks (VPNs)



Perimeter Defense

- Is it adequate?
 - Locating and securing all perimeter points is quite difficult
 - Less effective for large border
 - Inspecting/ensuring that remote connections are adequately protected is difficult
 - Insiders attack is often the most damaging



Firewalls

- Total isolation of networked systems is undesirable
 - Use firewalls to achieve selective border control
- Firewall
 - Is a configuration of machines and software
 - Limits network access
 - Come “for free” inside many devices: routers, modems, wireless base stations etc.
 - Alternate:
a firewall is a host that mediates access to a network, allowing and disallowing certain type of access based on a configured security policy



What Firewalls can't do

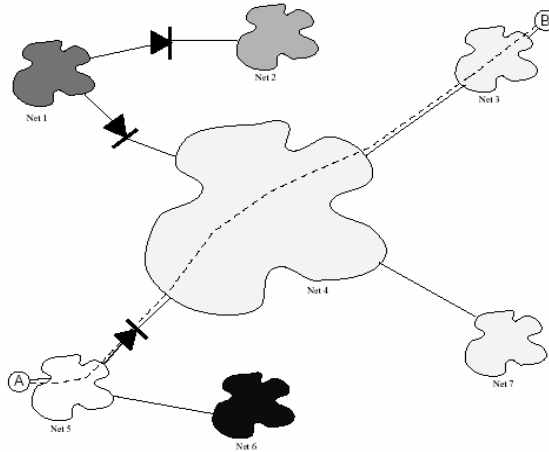
- They are not a panacea
 - Only adds to defense in depth
- If not managed properly
 - Can provide false sense of security
- Cannot prevent insider attack
- Firewalls act a particular layer (or layers)



What is a VPN?

- A network that supports a closed community of authorized users
- There is traffic isolation
 - Contents are secure
 - Services and resources are secure
- Use the public Internet as part of the virtual private network
- Provide security!
 - Confidentiality and integrity of data
 - User authentication
 - Network access control
- IPSec can be used

Tunneling in VPN



The Development of Firewalls First Generation

- Packet filtering firewalls
 - are simple networking devices that filter packets by examining every incoming and outgoing packet header
 - Can selectively filter packets based on values in the packet header, accepting or rejecting packets as needed
 - Can be configured to filter based on IP address, type of packet, port request, and/or other elements present in the packet

Packet Filtering Example Rules

Table 9-3 Packet Filtering Example Rules

Source Address	Destination Address	Service Port	Action
10.10.x.x	172.16.126.x	Any	Deny
192.168.x.x	10.10.x.x	Any	Deny
172.16.121.1	10.10.10.22	FTP	Allow
10.10.x.x	x.x.x.x	HTTP	Allow
x.x.x.x	10.10.10.25	HTTP	Allow
x.x.x.x	10.10.10.x	Any	Deny

Notes: These rules apply to a network at 10.10.x.x.

This table uses special, nonroutable IP addresses in the rules for this example. In reality, a firewall that connects to a public network will use real address ranges.

Second Generation

- Application-level firewalls
 - often consists of dedicated computers kept separate from the first filtering router (edge router)
 - Commonly used in conjunction with a second or internal filtering router - or proxy server
 - Proxy server, rather than the Web server, is exposed to outside world from within a network segment called the demilitarized zone (DMZ), an intermediate area between a trusted network and an untrusted network
- Application-level firewalls are implemented for specific protocols



Third Generation

- Stateful inspection firewalls,
 - keep track of each network connection established between internal and external systems using a state table
 - State tables track the state and context of each packet exchanged by recording which station sent which packet and when
 - can restrict incoming packets by allowing access only to packets that constitute responses to requests from internal hosts
 - If the stateful inspection firewall receives an incoming packet that it cannot match in its state table, then it uses ACL rights to determine whether to allow the packet to pass



Fourth Generation

- A fourth-generation firewall, or dynamic packet filtering firewall, allows only a particular packet with a specific source, destination, and port address to pass through the firewall
 - Does so by understanding how the protocol functions, and by opening and closing pathways in the firewall
- Dynamic packet filters are an intermediate form, between traditional static packet filters and application proxies



Firewall Architectures

- Each of the firewall generations can be implemented in a number of architectural configurations
- Four architectural implementations of firewalls are especially common:
 - Packet filtering routers
 - Screened-host firewalls
 - Dual-homed host firewalls
 - Screened-subnet firewalls



Packet Filtering Routers

- Most organizations with an Internet connection use some form of router between their internal networks and the external service provider
- Many of these routers can be configured to block packets that the organization does not allow into the network
- Such an architecture lacks auditing and strong authentication
 - Complexity of the access control lists used to filter the packets can grow to the point of degrading network performance

Packet Filtering Router/Firewall

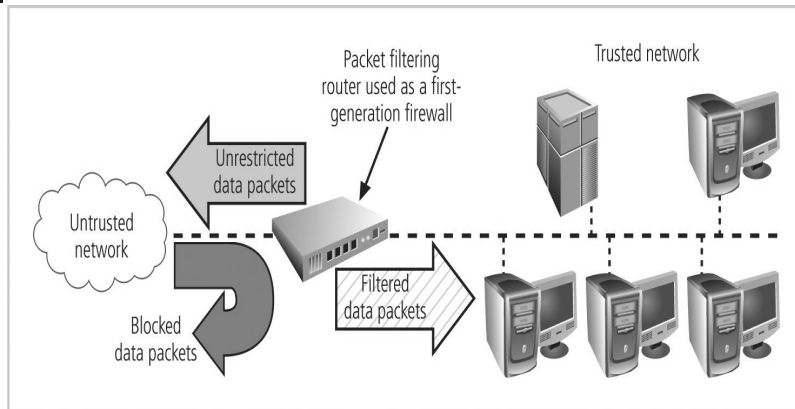


FIGURE 9-5 Packet Filtering Firewall

Screened-Host Firewall Systems

- Screened-host firewall systems
 - combine packet filtering router with a separate, dedicated firewall such as an application proxy server
 - allows the router to screen packets to minimize network traffic and load on the internal proxy
 - Application proxy examines an application layer protocol, such as HTTP, and performs the proxy services
 - This separate host, which is often referred to as a bastion host, represents a single, rich target for external attacks, and should be very thoroughly secured

Screened-Host Firewall

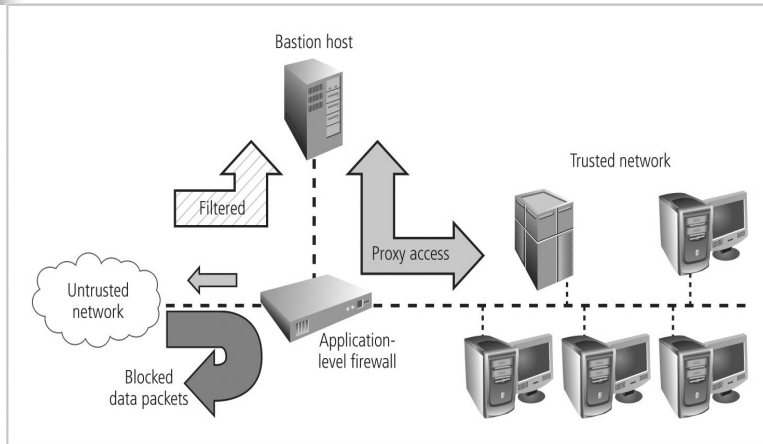


FIGURE 9-6 Screened-Host Firewall

Dual-Homed Host Firewalls

- In this configuration, the bastion host contains two network interfaces:
 - One connected to external network
 - One connected to internal network, requiring all traffic to travel through the firewall to move between the internal and external networks
- Network-address translation (NAT) is often implemented with this architecture
 - Converts external IP addresses to special ranges of internal IP addresses

Dual-Homed Host Firewalls (Continued)

- These special, non-routable addresses consist of three different ranges:
 - 10.x.x.x , > 16.5 million usable addresses
 - 192.168.x.x , > 65,500 addresses
 - 172.16.0.x - 172.16.15.x , > 4000 usable addresses

Figure 9-7 Dual-Homed Host Firewall

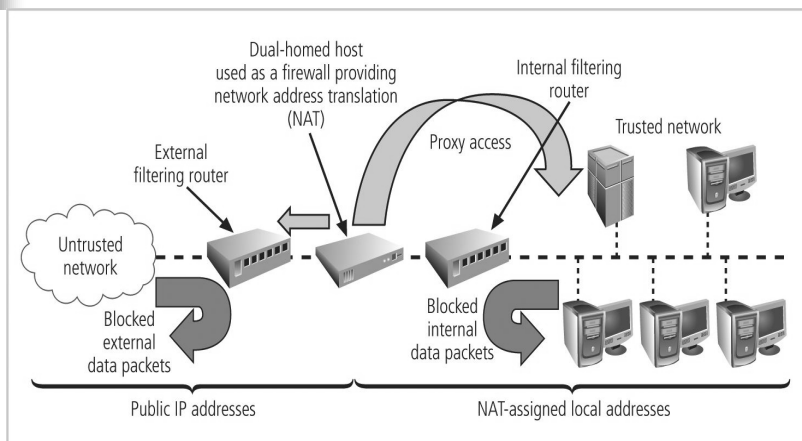




FIGURE 9-7 Dual-Homed Host Firewall



Screened-Subnet Firewalls (with DMZ)

- Screened-subnet firewall
 - consists of one or more internal bastion hosts located behind a packet filtering router, with each host protecting the trusted network
- First general model uses two filtering routers, with one or more dual-homed bastion hosts between them



Screened-Subnet Firewalls (with DMZ)

- Second general model (next slide) shows connections are routed as follows:
 - Connections from the outside or untrusted network are routed through an external filtering router
 - Connections from the outside or untrusted network are routed into—and then out of—a routing firewall to the separate network segment known as the DMZ
 - Connections into the trusted internal network are allowed only from the DMZ bastion host servers

Screened Subnet (DMZ)

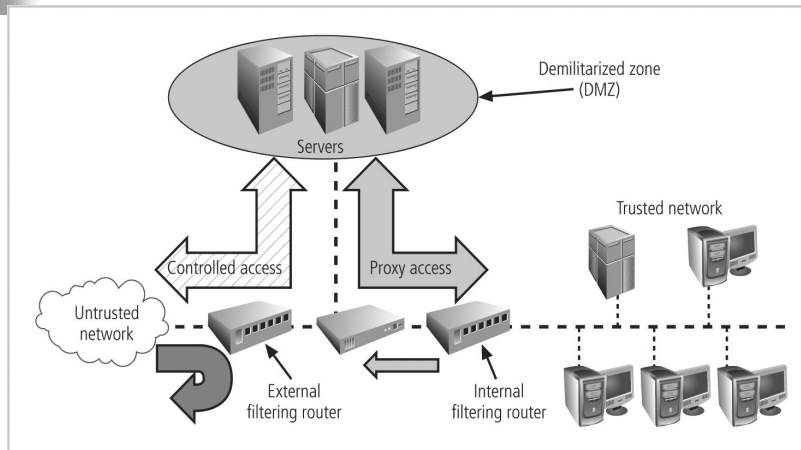


FIGURE 9-8 Screened-Subnet (DMZ)

Selecting the Right Firewall

- When evaluating a firewall, ask the following questions:
 - What type of firewall technology offers the right balance between protection and cost for the needs of the organization?
 - What features are included in the base price? What features are available at extra cost? Are all cost factors known?
 - How easy is it to set up and configure the firewall? How accessible are the staff technicians who can competently configure the firewall?
 - Can the candidate firewall adapt to the growing network in the target organization?



Managing Firewalls

- Any firewall device—
 - must have its own configuration that regulates its actions
- A policy regarding the use of a firewall should be articulated before it is made operable
- In practice, configuring firewall rule sets can be something of a nightmare
 - Each firewall rule must be carefully crafted, placed into the list in the proper sequence, debugged, and tested



Managing Firewalls

- Proper rule sequence ensures that the most resource-intensive actions are performed after the most restrictive ones, thereby reducing the number of packets that undergo intense scrutiny
- Firewalls:
 - Deal strictly with defined patterns of measured observation
 - Are prone to programming errors, flaws in rule sets, and other inherent vulnerabilities
 - Are designed to function within limits of hardware capacity
 - Can only respond to patterns of events that happen in an expected and reasonably simultaneous sequence



Firewall Best Practices

- All traffic from trusted network is allowed out
- Firewall device is never accessible directly from public network
- Simple Mail Transport Protocol (SMTP) data is allowed to pass through the firewall, but should be routed to a SMTP gateway
- All Internet Control Message Protocol (ICMP) data should be denied
- Telnet (terminal emulation) access to all internal servers from the public networks should be blocked
- When Web services are offered outside the firewall, HTTP traffic should be handled by some form of proxy access or DMZ architecture




Dial-Up Protection

- Attacker who suspects that an organization has dial-up lines can use a device called a war-dialer to locate connection points
- Network connectivity using dial-up connections is usually much simpler and less sophisticated than Internet connections
- For the most part, simple user name and password schemes are the only means of authentication



RADIUS and TACACS

- RADIUS and TACACS:
 - Systems that authenticate credentials of users trying to access an organization's network via a dial-up connection
- Typical dial-up systems place authentication of users on system connected to modems
 - Remote Authentication Dial-In User Service (RADIUS) system centralizes the management of user authentication
 - Places responsibility for authenticating each user in the central RADIUS server



RADIUS and TACACS (Continued)

- When a remote access server (RAS) receives a request for a network connection from a dial-up client
 - It passes the request along with the user's credentials to the RADIUS server
 - RADIUS then validates the credentials
- Terminal Access Controller Access Control System (TACACS) works similarly
 - Is based on a client/server configuration

Figure 9-9 RADIUS Configuration

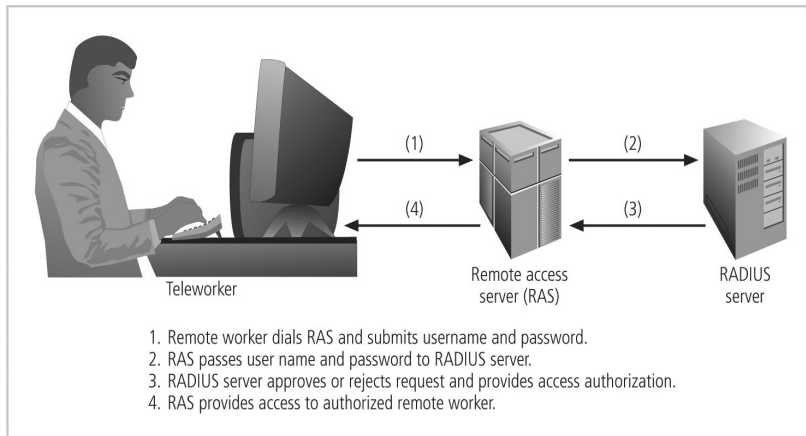


FIGURE 9-9 RADIUS Configuration

Managing Dial-Up Connections

- Organizations that continue to offer dial-up remote access must deal with a number of thorny issues:
 - Determine how many dial-up connections the organization has
 - Control access to authorized modem numbers
 - Use call-back whenever possible
 - Use token-based authentication if at all possible



Intrusion Detection



Intrusion Detection/Response

- Characteristics of systems not under attack:
 1. Actions of users/processes conform to statistically predictable patterns
 2. Actions of users/processes do not include sequences of commands to subvert security policy
 3. Actions of processes conform to specifications describing allowable actions
- Denning: Systems under attack fail to meet one or more of the these characteristics

Intrusion Detection

- Idea: Attack can be discovered by one of the above being violated
 - Automated attack tools
 - Designed to violate security policy
 - Example: *rootkits*: sniff passwords and stay hidden
- Practical goals of intrusion detection systems:
 - Detect a wide variety of intrusions (known + unknown)
 - Detect in a timely fashion
 - Present analysis in a useful manner
 - Need to monitor many components; proper interfaces needed
 - Be (sufficiently) accurate
 - Minimize *false positives* and *false negatives*

Figure 9-10 Intrusion Detection Systems

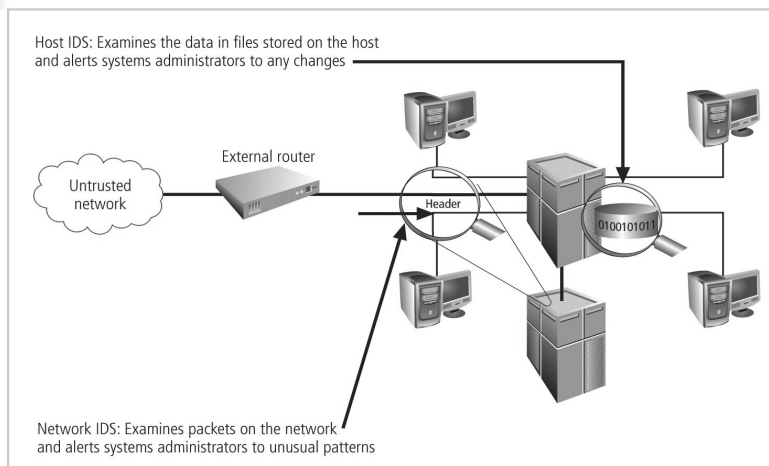


FIGURE 9-10 Intrusion Detection Systems



Host-Based IDS

- Host-based IDS works by configuring and classifying various categories of systems and data files
- In many cases, IDSs provide only a few general levels of alert notification
- Unless the IDS is very precisely configured, benign actions can generate a large volume of false alarms
- Host-based IDSs can monitor multiple computers simultaneously



Network-Based IDS

- Network-based IDSs
 - Monitor network traffic and, when a predefined condition occurs, notify appropriate administrator
 - Looks for patterns of network traffic
 - Must match known and unknown attack strategies against their knowledge base to determine whether an attack has occurred
 - Yield many more false-positive readings than do host-based IDSs
 - Because attempting to read network activity pattern to determine what is normal and what is not



IDS Types: Anomaly Detection

- Compare characteristics of system with expected values
 - report when statistics do not match
- Threshold metric: when statistics deviate from normal by threshold, sound alarm
 - E.g., Number of failed logins
- Statistical moments: based on mean/standard deviation of observations
 - Number of user events in a system
 - Time periods of user activity
 - Resource usages profiles
- Markov model: based on state, expected likelihood of transition to new states
 - If a low probability event occurs then it is considered suspicious



Statistical Anomaly-Based IDS


- Statistical anomaly-based IDS (stat IDS) or behavior-based IDS
 - First collects data from normal traffic and establishes a baseline
 - Then periodically samples network activity, based on statistical methods
 - Compares samples to baseline
 - When activity falls outside baseline parameters (known as the clipping level), IDS notifies the administrator
 - Advantage is that system is able to detect new types of attacks
 - Because it looks for abnormal activity of any type

Anomaly Detection: How do we determine normal?

- Capture average over time
 - But system behavior isn't always average
- Correlated events
 - Events may have dependencies
- Machine learning approaches
 - Training data obtained experimentally
 - Data should relate to as accurate normal operation as possible

IDS Types: Misuse Modeling

- Does sequence of instructions violate security policy?
 - Problem: How do we know all violating sequences?
- Solution: capture known violating sequences
 - Generate a rule set for an intrusion signature
 - But won't the attacker just do something different?
 - Often, no: *kiddie scripts, Rootkit, ...*
- Alternate solution: State-transition approach
 - Known "bad" state transition from attack (e.g. use petri-nets)
 - Capture when transition has occurred (user—root)



Signature-Based IDS

- Signature-based IDS or knowledge-based IDS
 - Examines data traffic for something that matches signatures which comprise preconfigured, predetermined attack patterns
 - Problem is that signatures must be continually updated, as new attack strategies emerge
 - Weakness is time frame over which attacks occur
 - If attackers are slow and methodical, they may slip undetected through the IDS, as their actions may not match a signature that includes factors based on duration of the events

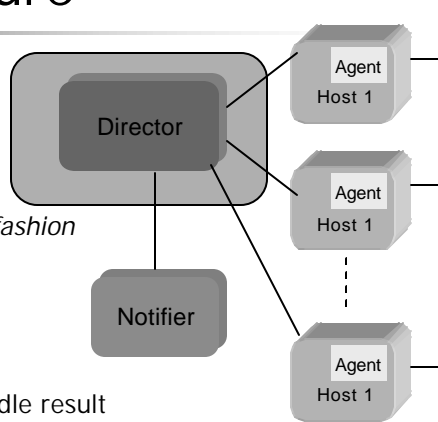


IDS Systems

- Anomaly Detection
 - Intrusion Detection Expert System (IDES) – successor is NIDES
 - Network Security Monitor (NSM)
- Misuse Detection
 - Intrusion Detection In Our Time- IDIOT (colored Petri-nets)
 - USTAT?
 - ASAX (Rule-based)
- Hybrid
 - NADIR (Los Alamos)
 - Haystack (Air force, adaptive)
 - Hyperview (uses neural network)
 - Distributed IDS (Haystack + NSM)

IDS Architecture

- Similar to Audit system
 - Log events
 - Analyze log
- Difference:
 - happens real-time - *timely fashion*
- (Distributed) IDS idea:
 - Agent generates log
 - Director analyzes logs
 - May be adaptive
 - Notifier decides how to handle result
 - GrIDS displays attacks in progress



Where is the Agent?

- Host based IDS
 - watches events on the host
 - Often uses existing audit logs
- Network-based IDS
 - Packet sniffing
 - Firewall logs



IDS Problem

- IDS useless unless accurate
 - Significant fraction of intrusions detected
 - Significant number of alarms correspond to intrusions
- Goal is
 - Reduce false positives
 - Reports an attack, but no attack underway
 - Reduce false negatives
 - An attack occurs but IDS fails to report



Intrusion Response

- Incident Prevention
 - Stop attack before it succeeds
 - Measures to detect attacker
 - Example: Jailing (also Honeypots)
 - Make attacker think they are succeeding and confine to an area
- Intrusion handling
 - Preparation for detecting attacks
 - Identification of an attack
 - Contain attack
 - Eradicate attack
 - Recover to secure state
 - Follow-up to the attack - Punish attacker



Containment

- Passive monitoring
 - Track intruder actions
 - Eases recovery and punishment
- Constraining access
 - Downgrade attacker privileges
 - Protect sensitive information
 - Why not just pull the plug?
 - Example: Honeypots



Eradication

- Terminate network connection
- Terminate processes
- Block future attacks
 - Close ports
 - Disallow specific IP addresses
 - Wrappers around attacked applications




Follow-Up

- Legal action
 - Trace through network
- Cut off resources
 - Notify ISP of action
- Counterattack
 - Is this a good idea?



Managing Intrusion Detection Systems

- IDSs must be configured using technical knowledge and adequate business and security knowledge to differentiate between routine circumstances and low, moderate, or severe threats
 - Properly configured IDS can translate a security alert into different types of notification
 - Poorly configured IDS may yield only noise
- Most IDSs monitor systems by means of agents, software that resides on a system and reports back to a management server



Managing Intrusion Detection Systems (Continued)

- Consolidated enterprise manager
 - Valuable tool in managing an IDS
 - Software that allows security professional to collect data from multiple host- and network-based IDSs and look for patterns across systems and subnetworks
 - Collects responses from all IDSs used to identify cross-system probes and intrusions



Vulnerability Analysis



Vulnerability Analysis

- Vulnerability or security flaw: specific failures of security controls (procedures, technology or management)
 - Errors in code
 - Human violators
 - Mismatch between assumptions
- Exploit: Use of vulnerability to violate policy
- Attacker: Attempts to exploit the vulnerability



Techniques for Detecting Vulnerabilities

- System Verification
 - Determine preconditions, post-conditions
 - Validate that system ensures post-conditions given preconditions
 - Can prove the absence of vulnerabilities
- Penetration testing
 - Start with system/environment characteristics
 - Try to find vulnerabilities
 - Can not prove the absence of vulnerabilities



System Verification

- What are the problems?
 - Invalid assumptions
 - Limited view of system
 - Still an inexact science
 - External environmental factors
 - Incorrect configuration, maintenance and operation of the program or system



Penetration Testing

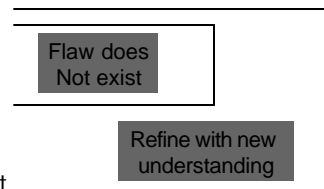
- Test strengths of security controls of the complete system
 - Attempt to violate stated policy
 - Works on in-place system
 - Framework for evaluating results
 - Examines procedural, operational and technological controls
- Typical approach: Red Team, Blue Team
 - Red team attempts to discover vulnerabilities
 - Blue team simulates normal administration
 - Detect attack, respond
 - White team injects workload, captures results

Types/layers of Penetration Testing

- Black Box (External Attacker)
 - External attacker has no knowledge of target system
 - Attacks often build on human element – Social Engineering
- System access provided (External Attacker)
 - Red team provided with limited access to system
 - Models external attack
 - Goal is to gain normal or elevated access
 - Then violate policy
- Internal attacker
 - Red team provided with authorized user access
 - Goal is to elevate privilege / violate policy

Red Team Approach Flaw Hypothesis Methodology:

- Information gathering
 - Examine design, environment, system functionality
- Flaw hypothesis
 - Predict likely vulnerabilities
- Flaw testing
 - Determine where vulnerabilities exist
- Flaw generalization
 - Attempt to broaden discovered flaws
- Flaw elimination (often not included)
 - Suggest means to eliminate flaw



Problems with Penetration Testing

- Nonrigorous
 - Dependent on insight (and whim) of testers
 - No good way of evaluating when “complete”
- How do we make it systematic?
 - Try all classes of likely flaws
 - But what are these?
- Vulnerability Classification!

Vulnerability Classification

- Goal: describe spectrum of possible flaws
 - Enables design to avoid flaws
 - Improves coverage of penetration testing
 - Helps design/develop intrusion detection
- How do we classify?
 - By how they are exploited?
 - By where they are found?
 - By the nature of the vulnerability?

Example flaw: xterm log

- xterm runs as root
 - Generates a log file
 - Appends to log file if file exists
- Problem: In /etc/passwd log_file
- Solution

```
if (access("log_file", W_OK) == 0)
    fd = open("log_file", O_WRONLY|O_APPEND)
```
- What can go wrong?

Example: Finger Daemon (*exploited by Morris worm*)

- finger sends name to fingerd
 - fingerd allocates 512 byte buffer on stack
 - Places name in buffer
 - Retrieves information (local finger) and returns
- Problem: If name > 512 bytes, overwrites return address
- Exploit: Put code in "name", pointer to code in bytes 513+
 - Overwrites return address



Vulnerability Classification: *Generalize*

- *xterm*: race condition between validation and use
- *fingerd*: buffer overflow on the stack
- Can we generalize to cover all possible vulnerabilities?



RISOS: Research Into Secure Operating Systems (Seven Classes)

1. Incomplete parameter validation
 - Check parameter before use
 - E.g., buffer overflow –
2. Inconsistent parameter validation
 - Different routines with different formats for same data
3. Implicit sharing of privileged / confidential data
 - OS fails to isolate processes and users
4. Asynchronous validation / inadequate serialization
 - Race conditions and TOCTTOU flaws
5. Inadequate identification / authentication / authorization
 - Trojan horse; accounts without passwords
6. Violable prohibition / limit
 - Improper handling of bounds conditions (e.g., in memory allocation)
7. Exploitable logic error
 - Incorrect error handling, incorrect resource allocations etc.

Protection Analysis Model Classes

- Pattern-directed protection evaluation
 - Methodology for finding vulnerabilities
- Applied to several operating systems
 - Discovered previously unknown vulnerabilities
- Resulted in two-level hierarchy of vulnerability classes
 - Ten classes in all

PA flaw classes

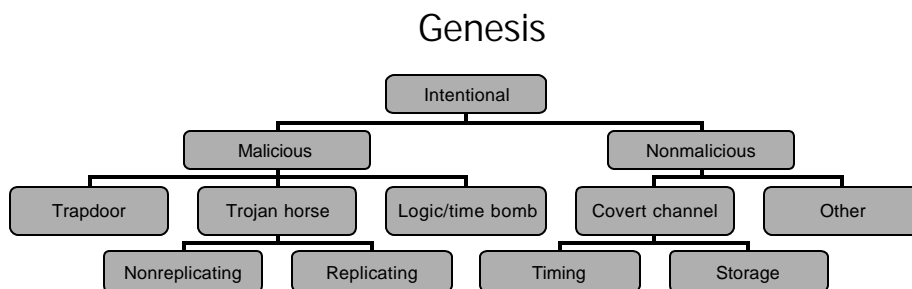
1. Improper protection domain initialization and enforcement
 - a. domain: Improper choice of initial protection domain
 - b. exposed representations: Improper isolation of implementation detail (Covert channels)
 - c. consistency of data over time: Improper change
 - d. naming: Improper naming (two objects with same name)
 - e. residuals: Improper deallocation or deletion
2. Improper validation of operands, queue management dependencies:
3. Improper synchronization
 - a. interrupted atomic operations: Improper indivisibility
 - b. serialization: Improper sequencing
4. critical operator selection errors: Improper choice of operand or operation

PA analysis procedure

- A pattern-directed protection evaluation approach
 - Collect known protection problems
 - Convert these problems to a more formalized notation (set of conditions)
 - Eliminate irrelevant features and abstract system-specific components into system-independent components (generalize raw patterns)
 - Determine relevant features of OS Code
 - Compare features with generic error patterns

NRL Taxonomy

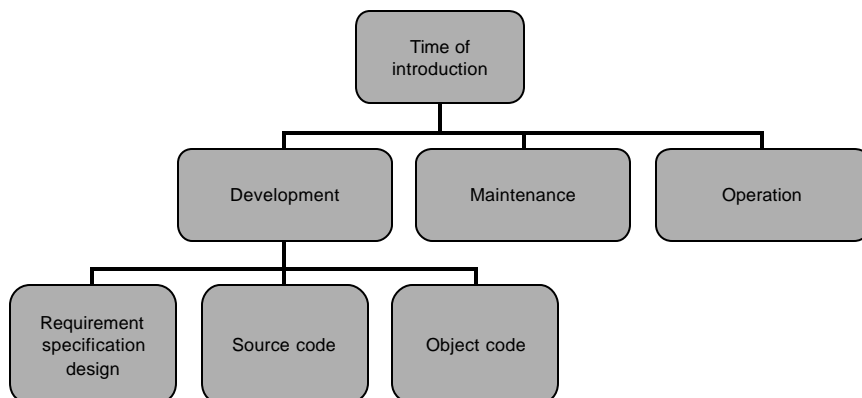
- Three classification schemes
 - How did it enter
 - When was it "created"
 - Where is it



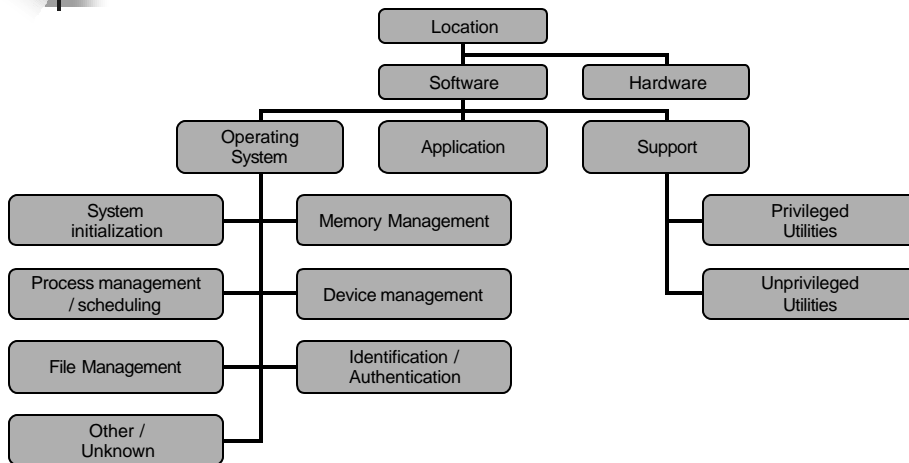
NRL Taxonomy (Genesis)

Inadvertent	Validation error (Incomplete/Inconsistent)
	Domain error (including object re-use, residuals, and exposed representation errors)
	Serialization/aliasing (including TCTTOU errors)
	Boundary conditions violation (including resource exhaustion and violable constraint errors)
	Other exploitable logic error

NRL Taxonomy: Time



NRL Taxonomy: Location



Aslam's Model

- Attempts to classify faults unambiguously
 - Decision procedure to classify faults
- Coding Faults
 - Synchronization errors
 - Timing window
 - Improper serialization
 - Condition validation errors
 - Bounds not checked
 - Access rights ignored
 - Input not validated
 - Authentication / Identification failure
- Emergent Faults
 - Configuration errors
 - Wrong install location
 - Wrong configuration information
 - Wrong permissions
 - Environment Faults

Common Vulnerabilities and Exposures (cve.mitre.org)

- Captures *specific* vulnerabilities
 - Standard name
 - Cross-reference to CERT, etc.
- Entry has three parts
 - Unique ID
 - Description
 - References

Name	CVE-1999-0965
Description	Race condition in xterm allows local users to modify arbitrary files via the logging option.
References	<ul style="list-style-type: none">•CERT:CA-93.17•XF:xterm

Buffer Overflow

- As much as 50% of today's widely exploited vulnerability
- Why do we have them
 - Bad language design
 - usually C, C++ : note they are good from other reasons
 - Hence good programming practice is needed
 - Java is a safer language
 - Poor programming

Buffer Overflow

- Some culprits
 - String operations that do no argument checking
 - strcpy() (most risky)
 - gets() (very risky)
 - scanf () (very risky)

```
void main(int argc, char **argv) {  
    char buf[256];  
    sscanf(argv[0], "%s", &buf)  
}
```

Buffer overflow if the input is more than 256 characters

```
Better design  
dst = (char *)malloc(strlen(src) + 1);  
strcpy(dst, src);
```

Auditing



What is Auditing?

- Logging
 - Recording events or statistics to provide information about system use and performance
- Auditing
 - Analysis of log records to present information about the system in a clear, understandable manner



Auditing goals/uses

- User accountability
- Damage assessment
- Determine causes of security violations
- Describe security state for monitoring critical problems
 - Determine if system enters unauthorized state
- Evaluate effectiveness of protection mechanisms
 - Determine which mechanisms are appropriate and working
 - Deter attacks because of presence of record



Problems

- What to log?
 - looking for violations of a policy, so record *at least* what will show such violations
 - Use of privileges
- What do you audit?
 - Need not audit everything
 - Key: what is the policy involved?



Audit System Structure

- Logger
 - Records information, usually controlled by parameters
- Analyzer
 - Analyzes logged information looking for something
- Notifier
 - Reports results of analysis



Logger

- Type, quantity of information recorded controlled by system or program configuration parameters
- May be human readable or not
 - If not, usually viewing tools supplied
 - Space available, portability influence storage format



Example: Windows NT

- Different logs for different types of events
 - System event logs record system crashes, component failures, and other system events
 - Application event logs record events that applications request be recorded
 - Security event log records security-critical events such as logging in and out, system file accesses, and other events
- Logs are binary; use event viewer to see them
- If log full, can have system shut down, logging disabled, or logs overwritten

Windows NT Sample Entry

Date: 2/12/2000 Source: Security
Time: 13:03 Category: Detailed Tracking
Type: Success EventID:592
User: WINDSOR\Administrator
Computer: WINDSOR

Description:

A new process has been created:

New Process ID: 2216594592

Image File Name:

 Program Files\Internet Explorer\IEXPLORE.EXE

Creator Process ID: 2217918496

User Name: Administrator

FDomain: WINDSOR

Logon ID: (0x0,0x14B4c4)

[would be in graphical format]

Analyzer

- Analyzes one or more logs
 - Logs may come from multiple systems, or a single system
 - May lead to changes in logging
 - May lead to a report of an event

- Using *swatch* to find instances of *telnet* from *tcpd* logs:
 /telnet/&!/localhost/&!/*.site.com/
- Query set overlap control in databases
 - If too much overlap between current query and past queries, do not answer
- Intrusion detection analysis engine (director)
 - Takes data from sensors and determines if an intrusion is occurring



Notifier

- Informs analyst, other entities of results of analysis
- May reconfigure logging and/or analysis on basis of results
- May take some action



Designing an Audit System

- Essential component of security mechanisms
- Goals determine what is logged
 - Idea: auditors want to detect violations of policy, which provides a set of constraints that the set of possible actions must satisfy
 - So, audit functions that may violate the constraints
- Constraint π_i : action — condition

Example: Bell-LaPadula

- Simple security condition and *-property
 - S reads O — $L(S) = L(O)$
 - S writes O — $L(S) = L(O)$
 - To check for violations, on each read and write, must log $L(S)$, $L(O)$, action (read, write), and result (success, failure)
 - Note: need not record S , O !
 - In practice, done to identify the object of the (attempted) violation and the user attempting the violation

Implementation Issues

- Show non-security or find violations?
 - Former requires logging initial state as well as changes
- Defining violations
 - Does “write” include “append” and “create directory”?
- Multiple names for one object
 - Logging goes by object and not name
 - Representations can affect this (if you read raw disks, you’re reading files; can your auditing system determine which file?)

Syntactic Issues

- Data that is logged may be ambiguous
 - BSM: two optional text fields followed by two mandatory text fields
 - If three fields, which of the optional fields is omitted?
- Solution: use grammar to ensure well-defined syntax of log files

Example Grammar

```
entry   : date host prog [ bad ] user [ "from" host ] "to" user "on" tty
date    : daytime
host    : string
prog    : string ":"
bad     : "FAILED"
user    : string
tty     : "/dev/" string
```

- Log file entry format defined unambiguously
- Audit mechanism could scan, interpret entries without confusion

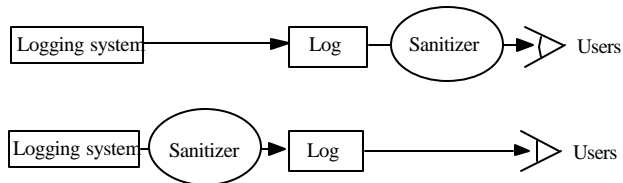
More Syntactic Issues

- Context
 - Unknown user uses anonymous *ftp* to retrieve file `"/etc/passwd"`
 - Logged as such
 - Problem: *which* `/etc/passwd` file?
 - One in system `/etc` directory
 - One in anonymous *ftp* directory `/var/ftp/etc`, and as *ftp* thinks `/var/ftp` is the root directory, `/etc/passwd` refers to `/var/ftp/etc/passwd`

Log Sanitization

- U set of users, P policy defining set of information $C(U)$ that U cannot see; log sanitized when all information in $C(U)$ deleted from log
- Two types of P
 - $C(U)$ can't leave site
 - People inside site are trusted and information not sensitive to them
 - $C(U)$ can't leave system
 - People inside site not trusted or (more commonly) information sensitive to them
 - Don't log this sensitive information

Logging Organization



- Top prevents information from leaving site
 - Users' privacy not protected from system administrators, other administrative personnel
- Bottom prevents information from leaving system
 - Data simply not recorded, or data scrambled before recording (Cryptography)

Reconstruction

- *Anonymizing sanitizer* cannot be undone
 - No way to recover data from this
- *Pseudonymizing sanitizer* can be undone
 - Original log can be reconstructed
- Importance
 - Suppose security analysis requires access to information that was sanitized?



Issue

- Key: sanitization must preserve properties needed for security analysis
- If new properties added (because analysis changes), may have to resanitize information
 - This *requires* pseudonymous sanitization or the original log



Example

- Company wants to keep its IP addresses secret, but wants a consultant to analyze logs for an address scanning attack
 - Connections to port 25 on IP addresses 10.163.5.10, 10.163.5.11, 10.163.5.12, 10.163.5.13, 10.163.5.14,
 - Sanitize with random IP addresses
 - Cannot see sweep through consecutive IP addresses
 - Sanitize with sequential IP addresses
 - Can see sweep through consecutive IP addresses



Generation of Pseudonyms

- Devise set of pseudonyms to replace sensitive information
 - Replace data with pseudonyms that preserve relationship
 - Maintain table mapping pseudonyms to data
- Use random key to encipher sensitive datum and use secret sharing scheme to share key
 - Used when insiders cannot see unsanitized data, but outsiders (law enforcement) need to
 - (t, n) –threshold scheme: requires t out of n people to read data



Application Logging

- Applications logs made by applications
 - Applications control what is logged
 - Typically use high-level abstractions such as:
su: bishop to root on /dev/tty0
 - Does not include detailed, system call level information such as results, parameters, etc.

System Logging

- Log system events such as kernel actions
 - Typically use low-level events

```
3876 ktrace CALL execve(0xbfbff0c0,0xbfbff5cc,0xbfbff5d8)
3876 ktrace NAMI "/usr/bin/su"
3876 ktrace NAMI "/usr/libexec/ld-elf.so.1"
3876 su RET execve 0
3876 su CALL __sysctl(0xbfbff47c,0x2,0x2805c928,0xbfbff478,0,0)
3876 su RET __sysctl 0
3876 su CALL mmap(0,0x8000,0x3,0x1002,0xffffffff,0,0,0)
3876 su RET mmap 671473664/0x2805e000
3876 su CALL geteuid
3876 su RET geteuid 0
```
 - Does not include high-level abstractions such as loading libraries (as above)


Contrast

- Differ in focus
 - Application logging focuses on application events, like failure to supply proper password, and the broad operation (what was the reason for the access attempt?)
 - System logging focuses on system events, like memory mapping or file accesses, and the underlying causes (why did access fail?)
- System logs usually much bigger than application logs
- Can do both, try to correlate them



Design

- *A posteriori* design
 - Need to design auditing mechanism for system not built with security in mind
- Goal of auditing
 - Detect *any* violation of a stated policy
 - Focus is on policy and actions designed to violate policy; specific actions may not be known
 - Detect actions *known* to be part of an attempt to breach security
 - Focus on specific actions that have been determined to indicate attacks



Detect Violations of Known Policy

- Goal: does system enter a disallowed state?
- Two forms
 - State-based auditing
 - Look at current state of system
 - Transition-based auditing
 - Look at actions that transition system from one state to another



State-Based Auditing

- Log information about state and determine if state is allowed
 - Assumption: you can get a snapshot of system state
 - Snapshot needs to be consistent
 - Non-distributed system needs to be quiescent



Example

- File system auditing tools (e.g. tripwire)
 - Thought of as analyzing single state (snapshot)
 - In reality, analyze many slices of different state unless file system quiescent
 - Potential problem: if test at end depends on result of test at beginning, relevant parts of system state may have changed between the first test and the last
 - Classic TOCTTOU flaw (time to check to time of use)




Transition-Based Auditing

- Log information about action, and examine current state and proposed transition to determine if new state would be disallowed
 - Note: just analyzing the transition may not be enough; you may need the initial state
 - Tend to use this when specific transitions *always* require analysis (for example, change of privilege)



Example

- TCP access control mechanism intercepts TCP connections and checks against a list of connections to be blocked
 - Obtains IP address of source of connection
 - Logs IP address, port, and result (allowed/blocked) in log file
 - Purely transition-based (current state not analyzed at all)



Detect Known Violations of Policy

- Goal: does a specific action and/or state that is known to violate security policy occur?
 - Assume that action *automatically* violates policy
 - Policy may be implicit, not explicit
 - Used to look for known attacks



Scanning Tools



Scanning and Analysis Tools

- Scanning and analysis tools can find vulnerabilities in systems, holes in security components, and other unsecured aspects of the network
- Conscientious administrators
 - Will have several informational web sites bookmarked
 - Frequently browse for new vulnerabilities, recent conquests, and favorite assault techniques
 - Nothing wrong with using tools used by attackers to examine own defenses and search out areas of vulnerability



Scanning and Analysis Tools

- Scanning tools collect the information that an attacker needs to succeed
- Footprinting
 - Organized research of the Internet addresses owned or controlled by a target organization
- Fingerprinting
 - Entails the systematic examination of all of the organization's network addresses
 - Yields a detailed network analysis that reveals useful information about the targets of the planned attack



Port Scanners

- Port
 - Network channel or connection point in a data communications system
- Port scanning utilities (or port scanners)
 - Can identify (or fingerprint) active computers on a network and active ports and services on those computers, the functions and roles fulfilled by the machines, and other useful information



Port Scanners (Continued)

- Well-known ports are those from 0 through 1023
- Registered ports are those from 1024 through 49151
- Dynamic and private ports are those from 49152 through 65535
- Open ports
 - Can be used to send commands to a computer
 - Gain access to a server
 - Exert control over a networking device
 - Thus must be secured

Commonly Used Port Numbers

Table 9-4 Commonly Used Port Numbers

Port Numbers	Description
20 and 21	File Transfer Protocol (FTP)
25	Simple Mail Transfer Protocol (SMTP)
53	Domain Name Services (DNS)
67 and 68	Dynamic Host Configuration Protocol (DHCP)
80	Hypertext Transfer Protocol (HTTP)
110	Post Office Protocol (POP3)
161	Simple Network Management Protocol (SNMP)
194	IRC Chat port (used for device sharing)
443	HTTP over SSL
8080	Proxy services

Vulnerability Scanners

- Vulnerability scanners
 - Variants of port scanners
 - Capable of scanning networks for very detailed information
 - Identify exposed user names and groups
 - Show open network shares
 - Expose configuration problems and other server vulnerabilities



Packet Sniffers

- Packet sniffer
 - Network tool that collects and analyzes packets on a network
 - Can be used to eavesdrop on network traffic
 - Must be connected directly to a local network from an internal location
- To use a packet sniffer legally, you must:
 - Be on a network that the organization owns, not leases
 - Be under the direct authorization of the network's owners
 - Have the knowledge and consent of users
 - Have a justifiable business reason for doing so




Content Filters

- Content filter
 - Effectively protects organization's systems from misuse and unintentional denial-of-service conditions
 - Software program or a hardware/software appliance that allows administrators to restrict content that comes into a network
 - Most common application is restriction of access to Web sites with non-business-related material, such as pornography
 - Another application is restriction of spam e-mail
 - Ensure that employees are using network resources appropriately




Trap and Trace

- Trap function
 - Describes software designed to entice individuals illegally perusing internal areas of a network
- Trace function
 - Process by which the organization attempts to determine the identity of someone discovered in unauthorized areas of the network or systems
 - If identified individual is outside the security perimeter, then policy will guide the process of escalation to law enforcement or civil authorities



Managing Scanning and Analysis Tools

- Vitally important that security manager be able to see organization's systems and networks from viewpoint of potential attackers
 - Should develop a program using in-house resources, contractors, or an outsourced service provider to periodically scan his or her own systems and networks for vulnerabilities with the same tools that typical hacker might use



Managing Scanning and Analysis Tools (Continued)

- Drawbacks to using scanners and analysis tools, content filters, and trap and trace tools:
 - Do not have human-level capabilities
 - Most function by pattern recognition —only handle known issues
 - Most are computer-based —prone to errors, flaws, and vulnerabilities of their own
 - Designed, configured, and operated by humans —subject to human errors
 - Some governments, agencies, institutions, and universities have established policies or laws that protect the individual user's right to access content
 - Tool usage and configuration must comply with explicitly articulated policy — policy must provide for valid exceptions