

ENTS 689i

**Lecture 11:
Infrastructure Defense**

Part III: Network Security

Part II: Outline

- November 20 (Today)
 - Network Architecture
 - Access Control
 - Firewalls
 - Intrusion Detection Systems
 - Anomaly Detection
 - Misuse Detection
 - Honeypots
 - Network Auditing
 - Homework/lab/review for test

Network Infrastructure Defense

- Focus on the infrastructure defense mechanisms (beyond protocols)
- Defense in depth
 - Multiple layers of protection mechanisms
 - Varying mechanisms
- Knowledge is your advantage (use it)
 - If the attacker understands your network infrastructure better than you, you will never win!

Network Architecture/Design

- Network architecture/design can have a significant effect on security
 - Implementation of mechanisms to support the policy
 - Planning/design are critical
 - Important that security is “built-in”
- Design principles (Saltzer and Schroeder)
 - 8 principles for the design and implementation of security mechanisms (common sense)
 - Simplicity – minimization of complexity
 - Restriction - access information it needs (limit power)

Design Principles (1)

- **Least Privilege**
 - A subject should be given only those privileges necessary to complete its task
- **Fail-Safe Defaults**
 - Unless a subject is given explicit access to an object, it should be denied access
- **Economy of Mechanism**
 - Security mechanisms should be as simple as possible
- **Complete Mediation**
 - All accesses to an object must be checked to ensure they are allowed

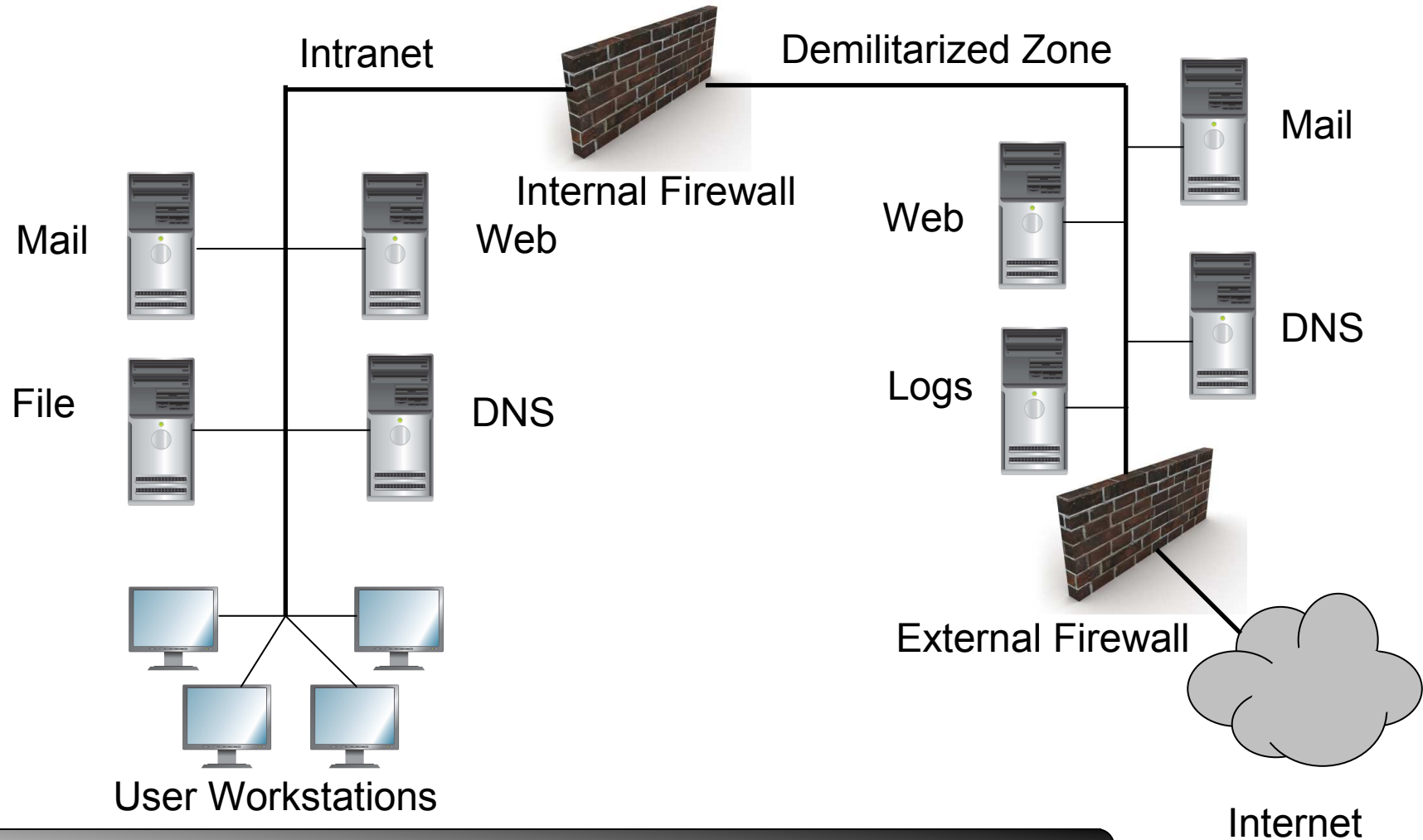
Design Principles (2)

- **Open Design**
 - Security of a mechanism should not depend on secrecy of its design or implementation
- **Separation of Privilege**
 - A system should not grant permission based on a single condition
- **Least Common Mechanism**
 - Mechanisms used to access resources should not be shared
- **Psychological Acceptability**
 - Security mechanisms should not make the resource more difficult to access than if security mechanisms were not present

Network Control Examples

- **Segmentation**
 - Networks (least privilege)
 - Services (least common mechanism)
 - Reduces the number of threats (exposure)
 - Limits the damage
- **Redundancy (least common mechanism)**
 - Duplicating critical components to increase reliability (failover)
 - Eliminate single points of failure
- **Complete mediation? separation of privilege?**

“Standard” Corporate Topology



Firewalls

- Construction industry: stop or slow progress of fire
 - Protecting the boundaries
- A security boundary between networks of differing trust and security levels by enforcing network level access control policy
 - A network reference monitor
 - Unbypassable, tamperproof, analyzable
 - Make decisions to allow or disallow passage of packets according to a specified policy (firewall policy)
 - Establish outer security wall/perimeter
 - Control point where security/audit can be imposed
 - Limit exposure
 - Partition the network (security domains)
 - Minimize damage
- Why is a firewall needed? (host security)

Firewall Policy

- **Firewalls enforce policies**
 - What kinds of data pass? blocked?
 - Security/non-security
 - Administrative boundaries
- **Firewall philosophies**
 - Default permit vs. default deny
 - Inbound filtering vs. outbound filtering
- **Firewall rules**
 - Rule generation
 - Manual
 - Automatic rule generation (interactive learning (ZA), passive observation)
 - Complexity

Types of Firewalls

- Firewall characteristics
 - Performance
 - Layers of the stack (context)
 - Stateless vs. stateful (resources)
 - Deployed location (visibility)
- Firewall types
 - Packet filters/firewalls
 - Stateful inspection firewalls
 - Application layer firewalls
 - Personal firewalls

Packet Filters

- Packet filters/firewalls
 - Routers ACLs
 - Individual packets (no context or connection state)
 - Allow or deny according to rules (Unidirectional)
- Filtering rules based on packet header information (fixed offsets)
 - IP addresses, ports, protocols, flags, interface
- High bandwidth links
- Challenges
 - Fragmentation, stateful filtering (sessions, FTP), UDP filtering

Example: Packet Filters

<u>Interface</u>	<u>Action</u>	<u>SrcAddr</u>	<u>SrcPort</u>	<u>DstAddr</u>	<u>DstPort</u>	<u>Flags</u>
Outside	block	128.168.1.1	*	*	*	*
Outside	allow	*	*	dmzmail	25	*
Inside	allow	dmzmail	25	*	*	*
Inside	allow	192.168.1.0/24	*	*	80	*
Inside	allow	192.168.1.0/24	*	*	80	*
Outside	allow	*	80	192.168.1.0/24	*	ACK
Outside	block	*	*	*	*	*

Stateful Inspection Filters

- Tracks state information of network connections
 - Maintains table of open connections
 - Passively monitors state of the connections
- Packets must match known connection state (efficient)
 - Sessions: TCP (IP addrs, port, sequence numbers, state, etc)
 - “Virtual Sessions”: UDP, ICMP (IP addrs, ports, type, etc)
 - Timeouts
- Context sensitive filtering decisions
 - Firewall rules
 - Context established by prior packets
 - Change filtering rules dynamically
- Network Address Translation (NAT)

Example: iptables

- **iptables** (<http://www.netfilter.org/>)
 - IP filtering firewall (Linux 2.4 and 1.6)
 - Chains
 - INPUT, OUTPUT, FORWARD, PREROUTING and POSTROUTING
- **Features**
 - Packet filtering, connection tracking (stateful), NAT, packet mangling, rate limiting, logging
- **Examples**
 - `iptables -A INPUT -i $IFACE -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT`
 - `iptables -A OUTPUT -o $IFACE -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT`

Application Layer Firewalls

- **Application proxy firewalls**
 - Client communicates through a mediator (proxy) to the server
 - Protection customized to the application
 - Leverage application context (access control)
 - Challenges: generality, latency, complexity
 - Examples: email, web
- **Deep packet inspection (extends stateful)**
 - Deep knowledge of the application payload
 - State of connection + application
 - Intrusion prevention systems (IDS + firewall)
 - “Ad-hoc MAC”

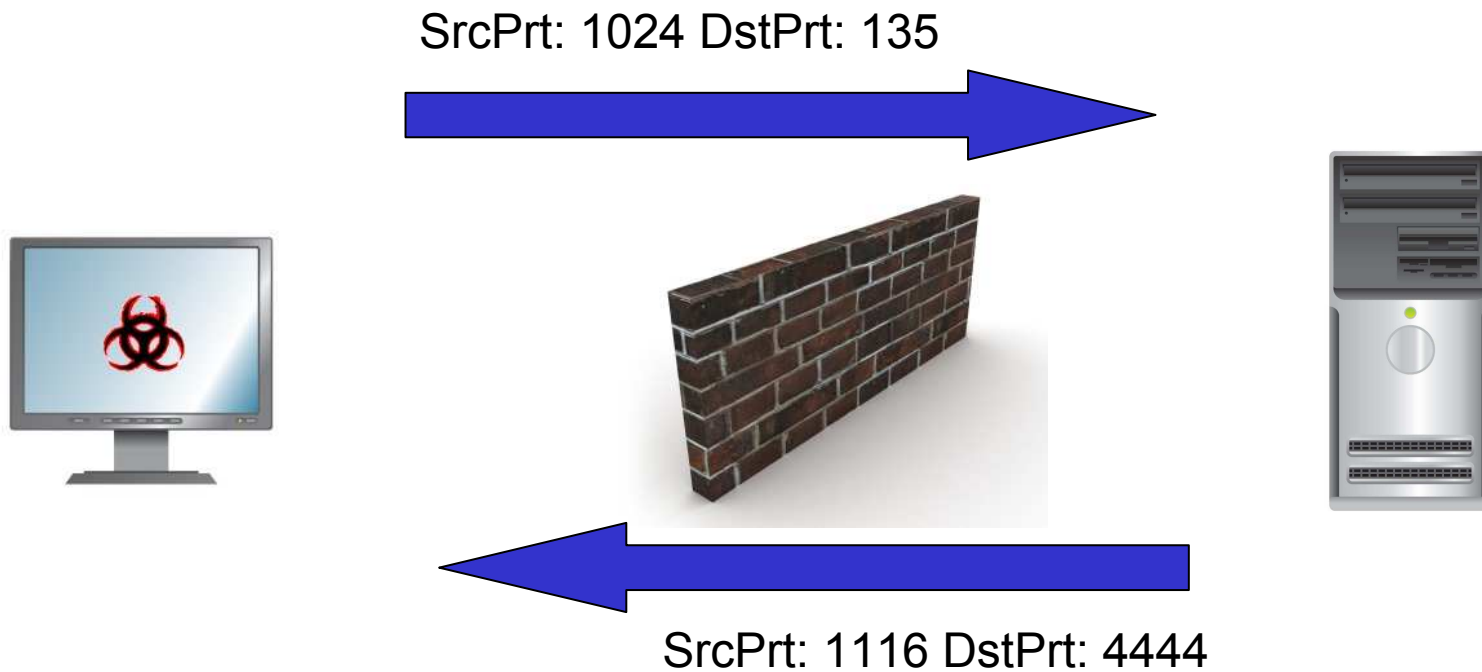
Personal Firewalls

- Intended to protect a single machine
 - Controls both in-bound/out-bound traffic
- Adaptive security policy
 - Deny/allow (programs)
- Advantages
 - Standalone protection (directly connected to the Internet)
 - Context (traffic → program), runtime awareness
- Disadvantages
 - Allow?
 - Performance, subversion, target of attack (Witty)

Firewalls Challenges

- Firewalls are not the panacea
 - Software bugs/ misconfiguration /evolve
- Insider threats
 - Hard outside, soft and chewy middle
 - Phishing attacks, VPNs, browser exploits
 - Mobile devices (laptops)
- Perimeter
 - Wireless, modems, cellular, business partners, etc
- Tunneling
 - http_tunnel, ssh tunnel, DNS tunnel, etc
- Reverse connections (shellcodes)

Example: Connect Back



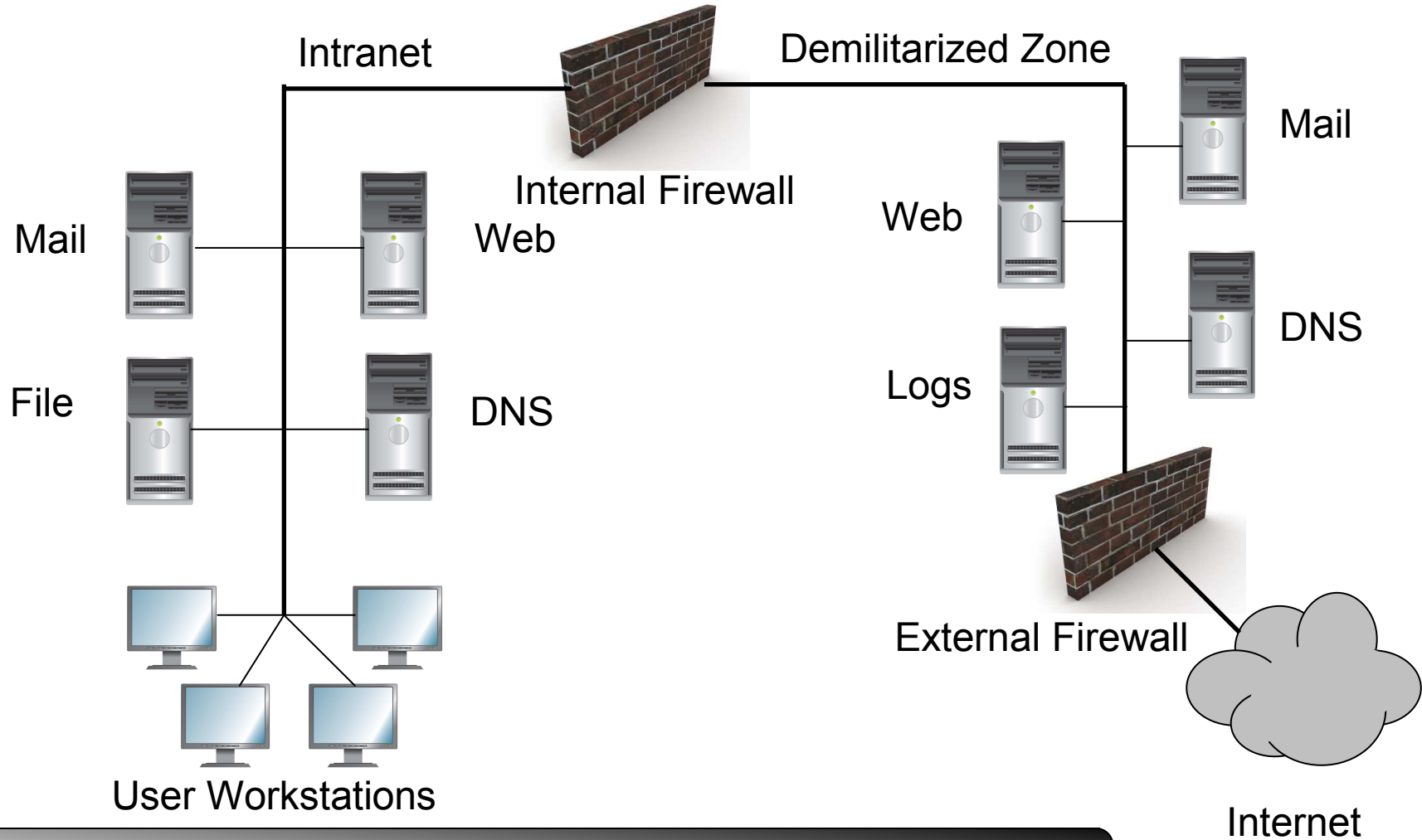
Intrusion Detection Systems

- Eventually protections (protocols, mechanisms, etc) will fail!
 - Vulnerabilities, threat model, assumptions (layered defenses)
- Intrusion detection systems
 - Monitor activity looking for signs of intrusion or intrusion attempts
 - Abnormal activity
 - Actionable alert (protect → detect → react)
- Where to monitor? (information sources: observables)
 - Host Intrusion Detection Systems (HIDS)
 - Network Intrusion Detection Systems (NIDS)
- Analysis types
 - Misuse
 - Anomaly

Network Intrusion Detection

- Network intrusion detection systems
 - Monitors network traffic for malicious or suspicious events
 - Software (promiscuous mode), network tap, span port
 - Deployed across the LAN (soft chewy middle)
- Why monitor from the network?
 - Low performance cost (host systems)
 - Transparent to users (adversary)
 - Visibility across the network
 - Stepping stone attacks/attack channels
 - Isolation
- Challenges?
 - Evasion techniques/resource attacks
 - Encryption

NIDS?



Alerts

- Alert accuracy is of critical importance
 - Base-rate fallacy
- True positive
 - IDS correctly alerts about attack
- False positive (type I error)
 - IDS reports attack when there is no attack
- True negative
 - IDS does not alert when there is no attack
- False negative (type II error)
 - IDS fails to identify an attack

Anomaly Detection

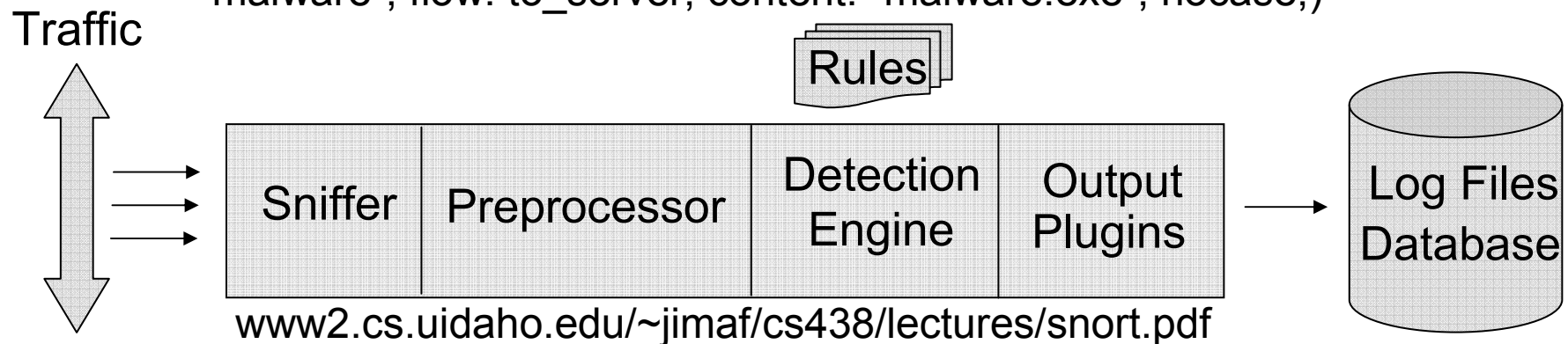
- Unexpected behavior as evidence of an intrusion
 - Builds a model of normal (training)
 - Thresholds/statistical modeling /Markov-based (state)
 - Deviations from normal
- Metrics
 - Characterize expected behavior of subjects (feature selection)
 - Amount of traffic, connection attempts, payload byte distributions, flows
- Advantages
 - Detect unknown attacks (broad coverage)
 - Evasion is challenging (know/learn model)
- Disadvantages
 - Training data, networks change (shift/drift), false positives (thresholds)
 - Actionable?

Misuse Detection

- Looks for activity known to be bad
 - Rule-based detection
 - Activity patterns that match a known attack or policy violation
 - Signatures (states, pattern-matching)
- Rule set
 - Knowledge of vulnerabilities/attacks (signatures)
- Advantages
 - Lower false positive rate
 - Specific alerts
- Disadvantages
 - Known attacks (narrow coverage)
 - Evasion
 - Constantly need to be updated

Example: Snort

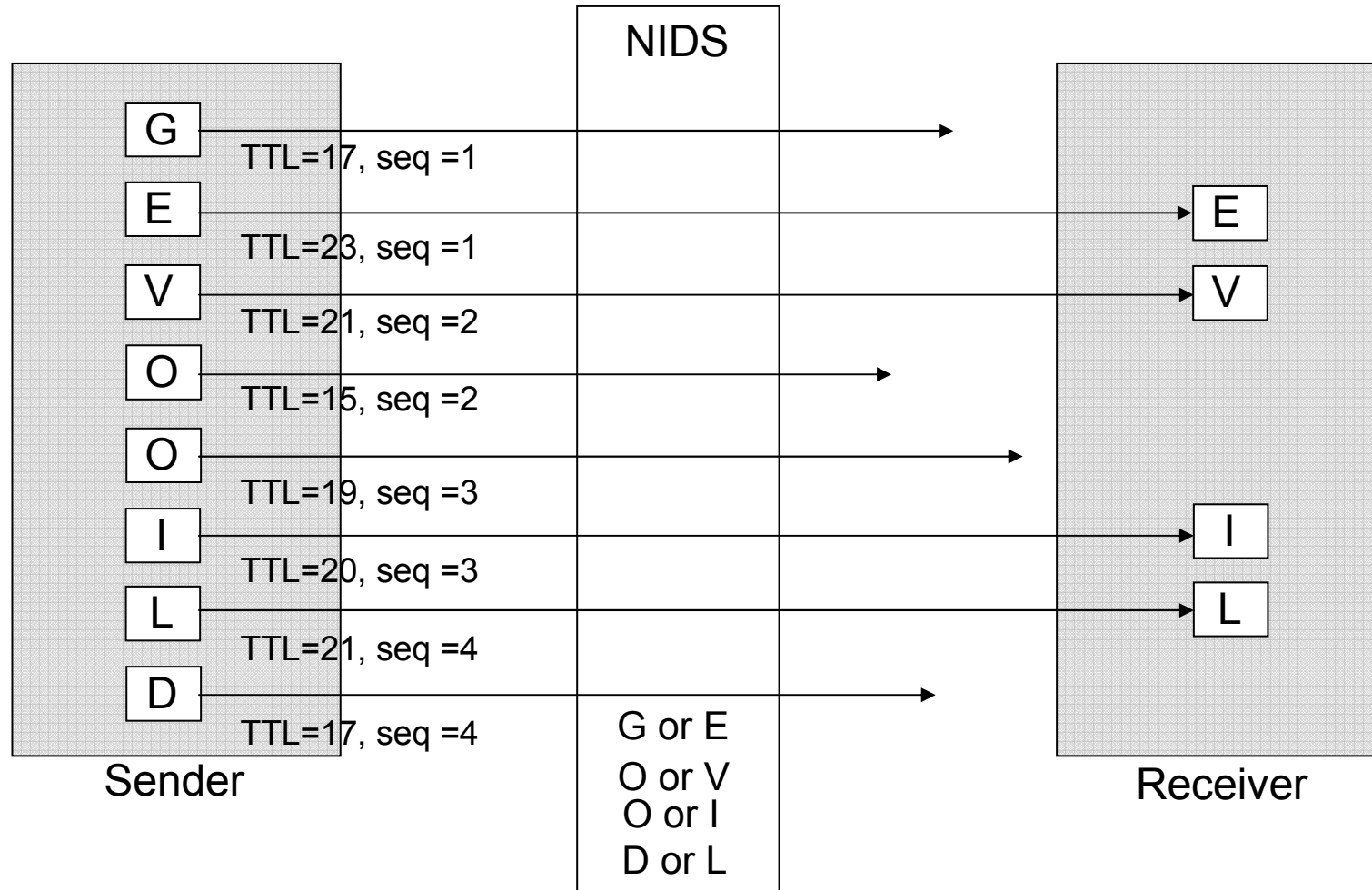
- Snort (<http://www.snort.org/>)
 - Open source signature based NIDS
 - Protocol analysis (TCP, UDP, ICMP)
- Preprocessors
 - Fragmentation, stream reassembly, etc.
- Example rule:
 - alert tcp \$EXTERNAL_NET any -> \$HTTP_SERVER 80 (msg:"Found malware"; flow: to_server; content: "malware.exe"; nocase;)



NIDS Evasion

- NIDS Evasion
 - Exploiting ambiguities in traffic stream seen by monitor
 - Layer 3/4 (Ptacek, Newsham, 1998) “Insertion and evasion”
 - Segmentation, fragmentation, TTL
 - Layer 7 (HTTP)
- Causes
 - Incomplete/inaccurate analysis capabilities
 - Incomplete knowledge of end-systems (protocol ambiguities)
 - Incomplete knowledge of network topology (TTL)
- Techniques for dealing with evasion
 - Inline normalization
 - Adapting algorithms based on target profiling
 - Bifurcating analysis

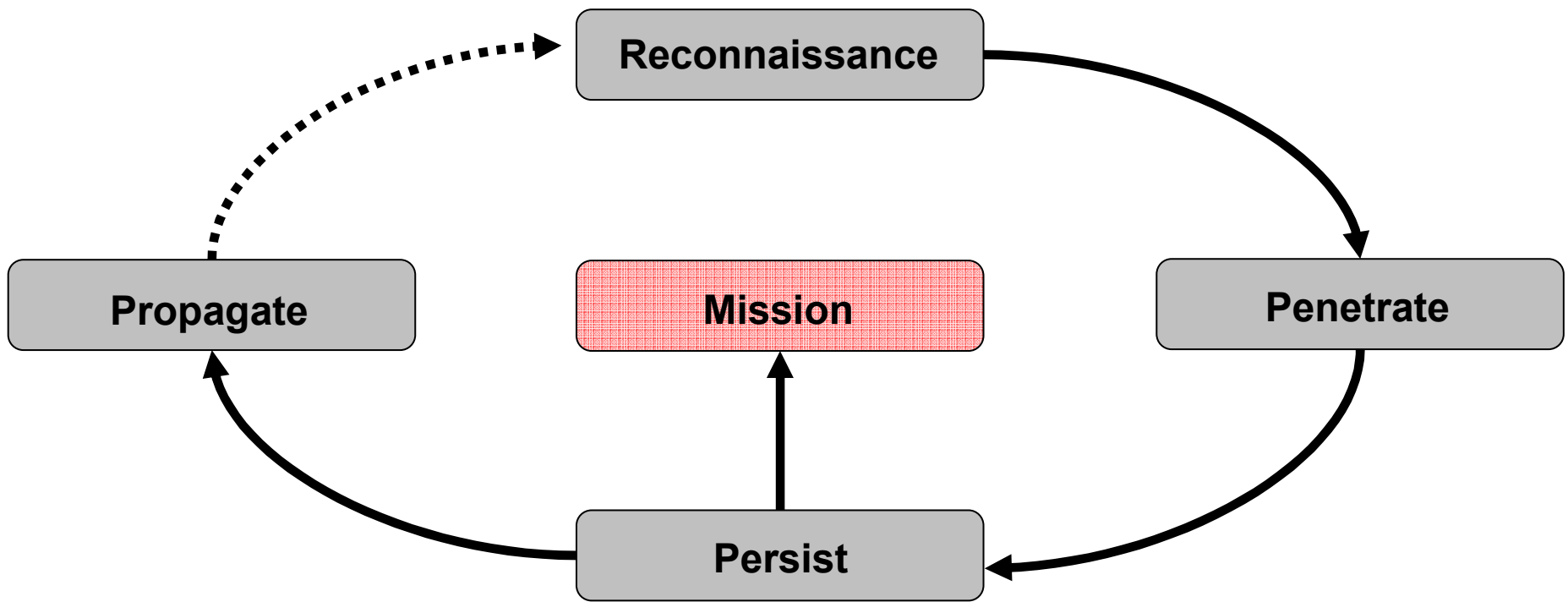
Example: Evasion



Deception as a Defense

- Deception for protection
 - Fred Cohen 1998 (<http://all.net/journal/deception/deception.html>)
 - Consume attacker resources (time, effort)
 - Force attackers to expose their techniques/mission
 - Force them to make a mistake that will be detected (create observables)
- Disrupt the attack cycle
 - Bad reconnaissance / intelligence
- Leverage knowledge disparity
- Deception Toolkit (DTK)
 - Make it appear as if the system has vulnerabilities

Disrupt the Attack



Honeypots

- “An information resource whose value lies in unauthorized or illicit use of that resource”
- Network intrusion detection system challenges
 - Encryption
 - False positives
 - Unknown attacks
- Study vulnerabilities and what an adversary does after gaining control (understand attacks)
 - Motivations, techniques, etc
- Different types of honeypots
 - High-interaction/Low-Interaction
 - Physical/Virtual

High-Interaction Honeypots

- Real systems and services
 - Computer, router, etc
- Attacker can completely compromise machine
 - Tools, tactics, mission
- Challenges
 - Maintenance, expense, scaling
 - Attack other systems
 - Detection/breakout
- Implementation
 - Physical machine
 - Virtualized – VMware, User-Mode Linux, Argos, etc

Example: Argos Honeypots

- Argos
 - QEMU: open source system emulator
 - <http://www.few.vu.nl/argos>
- Dynamic taint analysis
 - Trace the usage of network input
 - Detect zero-day attacks
- Generates attack footprint
 - Relevant registers
 - Memory footprint

Honeywall

- Platforms for further attacks (risk)
 - DDoS, stepping stones, etc
- Honeywall
 - Honeynet Project (<http://honeynet.org/tools/cdrom>)
 - Safeguard honeypots
 - Transparent bridge (data link layer)
- Capabilities
 - Data capture
 - Tcpdump, p0f, Argus, Sebek, etc
 - Data control
 - Rate limiting
 - “Extrusion” Prevention System (snort_inline)
 - Data analysis

Low-Interaction Honeypots

- Emulate aspects of a real machine
 - Services, network stacks, etc
 - Limited interaction
 - Statistics (i.e. Code Red)
- Advantages
 - Simplicity, maintenance (resources, etc)
 - Installation
 - Lower risk
- Examples
 - LaBrea, honeyd, Google Hack Honeypot (GHH), etc

Examples: Low Interaction

- **LaBrea (tarpit)**
 - Slowing down attackers (worms, etc)
 - Throttling (small window size)
 - Persistent capture (window size 0)
- **Honeyd (Dr. Niels Provos)**
 - Framework for virtual honeypots and network services
 - Instrument unallocated IP addresses
 - Scalability (hundreds of networks/thousands of hosts)
 - Operating system personalities
 - Nmap, Xprobe
 - Arbitrary routing topologies
 - Latency, packet loss, bandwidth

Honeytokens

- A digital entity whose value lies in its unauthorized use
 - Not just a computer
- Interaction represents unauthorized or malicious activity
 - Insider threat
 - Exfiltrate
- Examples
 - Credit card number, Office document, database entry, celebrity medical records
- Detect access
 - NIDS

Network Auditing

- Important to understand the state of your network
 - Network architecture
 - Information assets (PII, CCN, Medical Records)
 - Network defenses
 - Security weaknesses/exposures
- Large networks constantly evolving
 - Devices being added/reinstalled
 - Applications being deployed
 - Networks expanding (partnerships, wireless)
- New vulnerabilities constantly being found
- Proactively verify network security
 - Compliance

Network Mapping/Scanning

- Network Mapping
 - Finding unknown hosts/devices connected to the network (inventory)
 - Verifying operating system versions (OS fingerprinting)
 - Measuring uptime (time since reboot (patch))
 - Techniques
 - Scanning, passive
- Port Scanning
 - Searching your network for accessible/listening ports
 - Validate firewall configurations
- Tools
 - Nmap (“Network Mapper”) <http://nmap.org/>
 - P0f (“Passive OS Fingerprinting”) <http://lcamtuf.coredump.cx/p0f.shtml>

Vulnerability Scanning

- Vulnerability Scanning
 - Proactively looking for potential exposures and weaknesses
 - Devices open to known vulnerabilities
 - Vantage points/threat models
 - Generate reports and recommendations
 - Mitigate the threat (patches, updates, etc)
 - Regular vulnerability scanning
- Penetration testing
- Tools
 - Nessus (<http://www.nessus.org/nessus/>)
 - Metasploit (<http://www.metasploit.com/>)

Wireless Auditing

- Determining wireless perimeter
- Detecting access points
 - Rogue access points
 - Open access points
- Weak/vulnerable encryption
- Detecting wireless clients?
- Tools
 - Kismet (<http://www.kismetwireless.net/>)
 - Aircrack-ng (<http://www.aircrack-ng.org/doku.php>)

Packet Sniffing

- Intercept or log traffic traversing the network
 - Promiscuous mode interface
 - Pcap files
 - Protocol analysis
- Learn about the traffic on your network
 - Passwords
 - Sensitive information
 - Flows
- Tools
 - Wireshark (<http://www.wireshark.org/>)
 - Argus (<http://qosient.com/argus/>)
 - Dsniff (<http://www.monkey.org/~dugsong/dsniff/>)

References

- **Cryptography and Network Security**
 - William Stallings
- **Computer Security: Art and Science**
 - Matt Bishop
- **Security in Computing**
 - Charles P. Pfleeger and Shari Lawrence Pfleeger
- **Virtual Honeypots**
 - Niels Provos and Thorsten Holz