CSE 410/565 Computer Security
Spring 2022

Lecture 20: Intrusion Prevention

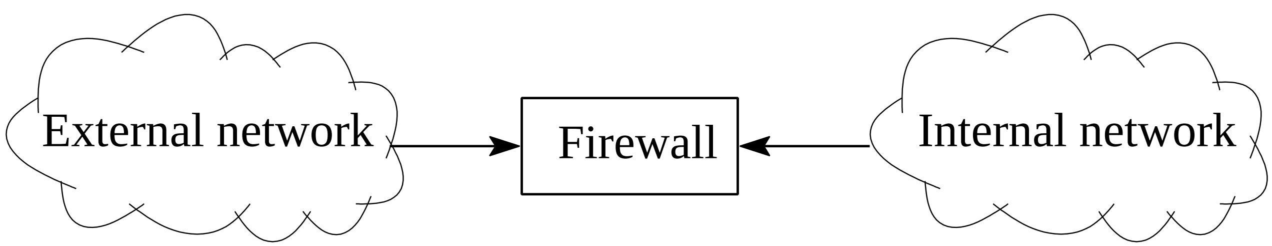
Department of Computer Science and Engineering
University at Buffalo

Lecture Overview

- Firewalls
  - purpose
  - types
  - locations
- Network perimeter security
- Defense in depth

Firewalls

- A firewall is security software that filters out unwanted or potentially dangerous traffic
  - a firewall can be used to protect a network from the outside world
    - external network (e.g., Internet) is considered to be untrusted
    - firewall is used to implement and enforce a security policy
  - it serves as a single protection point for entire enterprise
    - security management becomes easier
  - filtering can be done in both directions (with different rules)



Firewalls

- What can we expect from a firewall?
  - single point that blocks unauthorized users from the protected network and simplifies security management
  - monitoring and reporting of security-related events
  - implementation of virtual private networks by means of IPsec, tunneling
  - convenient place for integration of other functions for network management
- A firewall does not protect against attacks that don't go through the firewall
  - e.g., wireless connections, internal attacks, external devices connected directly to the internal machines/network

Firewalls

- Where can a firewall reside?
  - on a router
  - on a dedicated machine
  - personal firewall on a host
    - software that protects a single host rather than a network
    - e.g., Windows firewall, iptables in Linux, etc.
    - typically is configured to block most incoming traffic, but some applications can be let through
    - can be bypassed/disabled if host is compromised
- A firewall must be immune to penetration
  - ideally, it should run on a hardened system with a secured OS

Firewalls

- Types of firewalls
  - packet filtering
    - simplest kind of firewall
    - router has a list of access control rules
    - router checks each received packet against security rules to decide whether to forward or drop it
    - each rule specifies which packets it applies to based on a packet's header fields
      - can specify source and destination IP addresses, port numbers, protocol names, or wild cards
      - actions are `ALLOW` or `DROP`
      - $\langle$ACTION$\rangle$ $\langle$PTRCL$\rangle$ $\langle$SRC:PT$\rangle$ $\rightarrow$ $\langle$DEST:PT$\rangle$

Firewalls

- Packet filtering (cont.)
  - list of rules is examined one-by-one
  - first matching rules determines how packet will be handled
  - if no match is found, the default option can be to allow or drop
    - if the default option is drop, it is more noticeable to users
    - additional rules are added over time
    - this option, however, is preferred from security management point of view

Packet Filtering

- Policies based on IP header fields
  - a TCP or UDP service is specified by machine's IP address and port number
    - e.g., web server `engineering.buffalo.edu` is at 128.205.201.56 port 80
  - identify each service with triplet `(addr, prot, port)`
    - `addr` is machine's IP address (`a.b.c.d/[mask]`)
    - `prot` is TCP/UDP protocol identifier
    - `port` is the port number
  - example: all official web servers are located on subnet 12.34.56.x
    - add (12.34.56.0/24, TCP, 80) to allowed list

Packet Filtering

- Let's examine a sample ruleset
  - ```
    drop TCP *:* -> *:23
    allow *  *:* -> *:*
    ```
  - what does it do?
    -
    -
  - is this ruleset satisfactory?
    - there is no notion of a connection, inbound vs outbound connections
    - inbound and output packets to port 23 are dropped
    - default allow policy is undesirable
    ```

Packet Filtering

- Another example
  - assume that we want to allow
    - inbound connections to web server 12.34.56.78 on port 80
    - all outbound connections
    - nothing else
  - we create the following ruleset
    - ```
      allow TCP *:* -> 12.34.56.78:80
      allow TCP (our-hosts):* -> *:*
      drop   *  *:* -> *:*
      ```
  - there are problems with it
    - TCP connections are bidirectional, data have to be able to go both ways
      ```

Packet Filtering

- Recall that TCP handshake is 3-way
  - send SYN, receive SYN-ACK, send ACK, then send data with ACK
- Suppose an inside host connects to an external machine on port 25 (mail)
  - initial packets get through (using rule 2)
  - SYN-ACK is dropped (fails the first two rules, matches the last)
- We need to distinguish between two types of inbound packets
  - allow inbound packets associated with an outbound connection
  - disallow inbound packets associated with an inbound connection

Packet Filtering

- We use TCP feature to make this distinction
  - ACK bit is set on all packets except the first one
  - recipients discard any TCP packet with ACK bit if it is not associated with an existing TCP connection
- Revised ruleset
  - ```
    allow TCP *:* -> 12.34.56.78:80
    allow TCP (our-hosts):* -> *:*
    allow TCP *:* -> (our-hosts):* (if ACK bit set)
    drop   *   *:* -> *:*
    ```
  - rules 1 and 2 permit inbound connections to 12.34.56.78 port 80
  - rules 2 and 3 allow outbound connections to any port
    ```

Packet Filtering

- Let's see how our firewall stops packets
  - attacker wants to exploit finger service vulnerability (TCP port 79)
  - attempt 1: attacker sends SYN packet to internal machine
    - packet doesn't have ACK bit set, so firewall rule drops it
  - attempt 2: attacker sends SYN-ACK packet to internal machine
    - firewall permits the packets, but then it is dropped by the TCP stack (i.e., ACK bit set, but it is not part of an existing connection)
- We can customize the ruleset to let any types of packets through according to the policy
- Does it mean we done now? how about spoofed addresses?

Packet Filtering

- Suppose an attacker can spoof source IP address and performs the following attack
  - let 12.34.56.77 be an internal host
  - attacker sends a spoofed TCP SYN packet from address 12.34.56.77 to another internal machine on port 79
    - rule 2 in the ruleset allows the packet
  - target machine replies with SYN-ACK packet to 12.34.56.77 and waits for ACK (to finish handshake)
  - attacker sends spoofed ACK packet
  - attacker sends data packet(s)

Packet Filtering

- The attack above permits connections to internal hosts
  - it violates our security policy
  - it allows an attacker to exploit security vulnerabilities in internal machines
  - one difficulty: the attacker has to guess initial sequence number set by target in SYN-ACK packet to 12.34.56.77
    - the attacker doesn't see the response packet, but guessing might not be difficult
- What do we do now?
  - solve this by taking the interface a packet is coming from into consideration
  - mark a packet with interface id and incorporate ids into the rules

Packet Filtering

- New ruleset
  - internal interface is `in`, external interface `out`
  - ```
    allow TCP *:*/out -> 12.34.56.78:80/in
    allow TCP *:*/in -> *:*/out
    allow TCP *:*/out -> *:*/in (if ACK bit set)
    drop   *  *:* -> *:*
    ```
  - this allows inbound packets only to 12.34.56.78:80 (rule 1) or if ACK bit set (rule 3)
  - all other inbound packets are dropped
- Simple modification cleanly defeats IP spoofing threat
  - it simplifies ruleset administration (no need to hardcode internal hosts)
    ```

Other Types of Firewalls

- Stateless packet filtering has its limitations
  - small fragment attacks
    - TCP header can be split among several tiny IP packets
    - the hope is to circumvent filtering rules based on TCP fields
    - the easiest solution is to drop all packets that don't contain enough information in the first fragment
  - inability to recognize connections
    - most traffic is two-way
  - inability to examine upper-layer data and prevent application-specific attacks
  - inability to support advanced user authentication

Other Types of Firewalls

- Stateful packet inspection
  - packet decision is made in the context of a connection
  - if a packet is a new connection, check against security policy
  - if a packet is part of an existing connection, find it in the state table and update the table
    - this can be viewed as packet filtering with rules dynamically updated
- Example connection state table

source address	source port	dest address	dest port	conn state
219.22.123.32	2112	124.33.44.5	80	established
124.33.44.129	1030	132.65.89.2	80	established
124.33.44.7	1035	190.3.15.4	25	established

Other Types of Firewalls

- Application layer firewalls (or proxy firewalls)
  - is used as a relay for connections: Client $\leftrightarrow$ Proxy $\leftrightarrow$ Server
  - understands specific applications
    - limited versions of applications are available
    - proxy "impersonates" both sides of a connection
    - tends to be more secure than simple packet filters (can block application-specific attacks, can support authentication)
  - is resource-intensive (i.e., one process per connection)
  - certain proxies (e.g., HTTP) may cache data (e.g., web pages)

Firewall Location

- Firewall location
  - a firewall can be placed at different locations within a network
  - multiple firewalls can be used
- It is very common to have a firewall at the boundary of the entire network
- Subnets (especially with sensitive information and services) might have additional firewall(s)
- Finally, individual hosts might run firewall elements

Network Perimeter Security

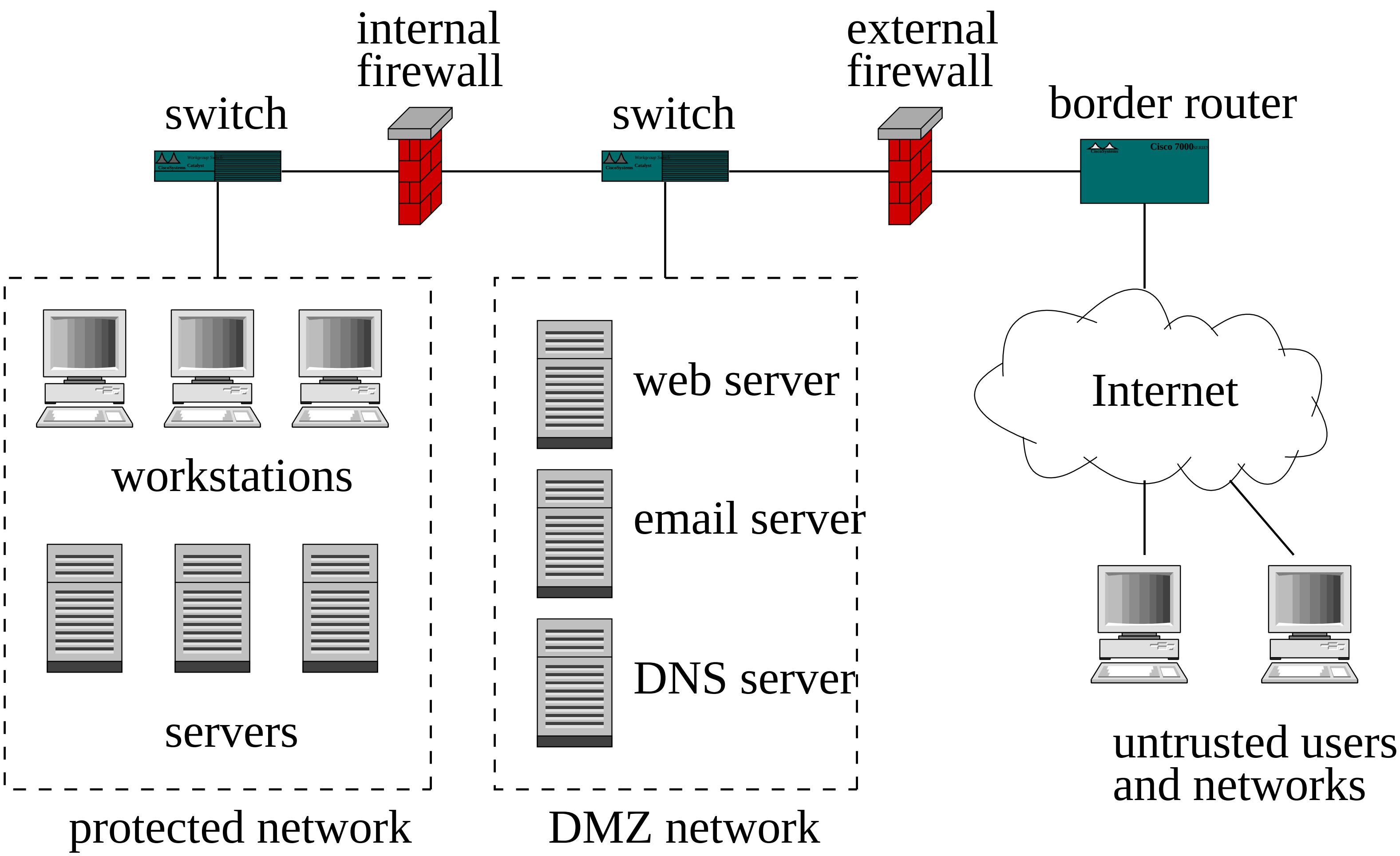
- We often want to have fortified boundary of our network
- The idea is to secure a small number of entry points into the network
  - similar concept is used in airports
- Tools we can use
  - border router
    - the last router you control before untrusted network (i.e., Internet)
  - firewall
    - a chokepoint device that decides what traffic is allowed
  - intrusion detection systems
    - an alarm system that detects malicious events and alerts administrators

Network Perimeter Security

- Tools we can use (cont.)
  - intrusion prevention system
    - inline IDS
    - provides automatic defense without administrators' involvement
  - demilitarized zone (DMZ)
    - small network providing public services
    - not as well protected as the rest of the network
      - there is often a firewall between DMZ and Internet
      - there is also a firewall between DMZ and internal network

Network Perimeter Security

- Firewall configuration with DMZ

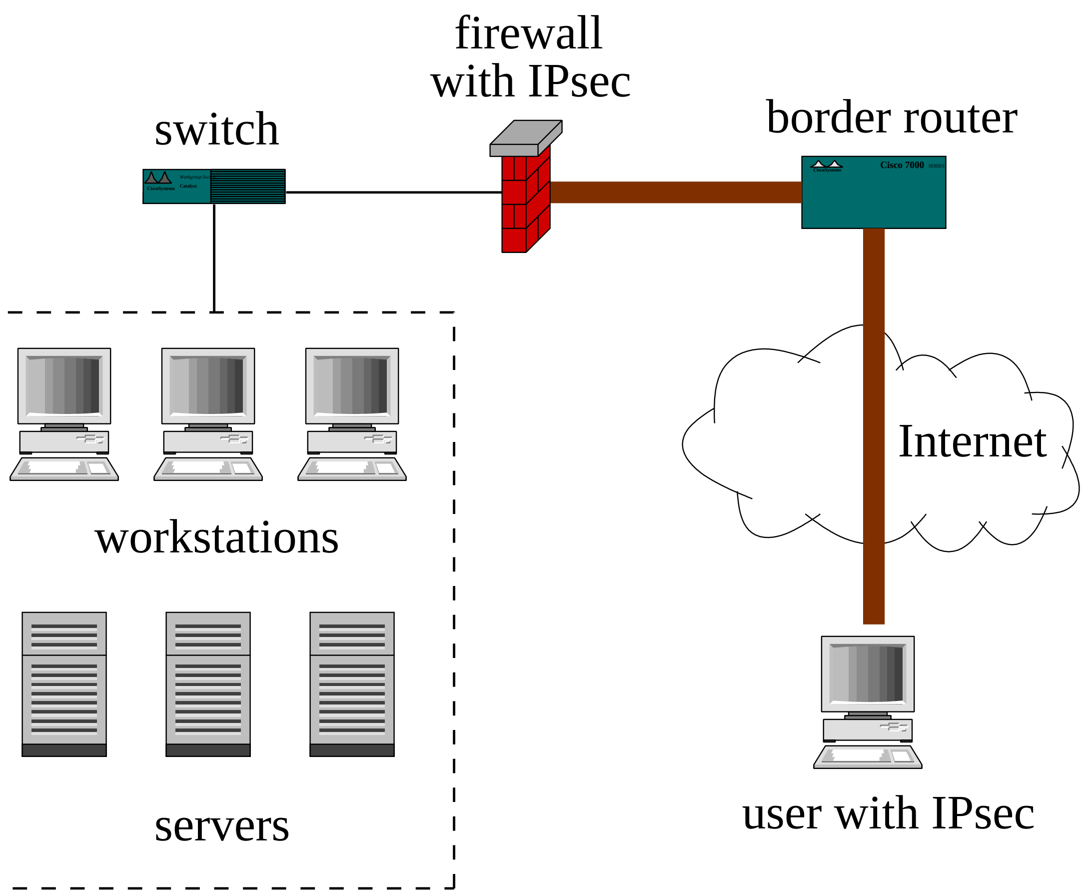


Network Perimeter Security

- Tools we can use (cont.)
  - virtual private network (VPN)
    - a protected network session formed across an unprotected channel such as Internet
    - hosts that connect through a VPN are part of the trusted network
    - a secure tunnel can be formed using IPsec
      - a user who is away from her network encrypts her connections and forwards them across the internet
      - a firewall at the boundary of home network decrypts traffic
    - user gets to use internal resources as she was on the internal network

Network Perimeter Security

- Illustration of VPN



Defense in Depth

- Defense in depth
  - security strategy that consists of layers of defense placed at various points in the enterprise
  - addresses vulnerabilities in all of technology, personnel, and operations of a system
- Defense in depth components
  - perimeter
    - static packet filter, stateful firewall, proxy firewall, IDS and IPS, VPN device
  - internal network
    - ingress and egress filtering on every router, internal firewalls, IDS sensors

Defense in Depth

- Defense in depth components (cont.)
  - individual hosts
    - host-centric firewalls
    - anti-virus software
    - configuration management
    - audit
  - human factor
    - user education
    - training
    - appropriate privilege assignment

Conclusions

- Now we have a global picture of network and systems protection
  - anti-virus software
  - intrusion detection systems
  - intrusion prevention systems
  - firewalls
  - audit
  - training