PennState

CSE543 - Computer and Network Security

Module: Firewalls

Asst. Prof. Syed Rafiul Hussain

Problem

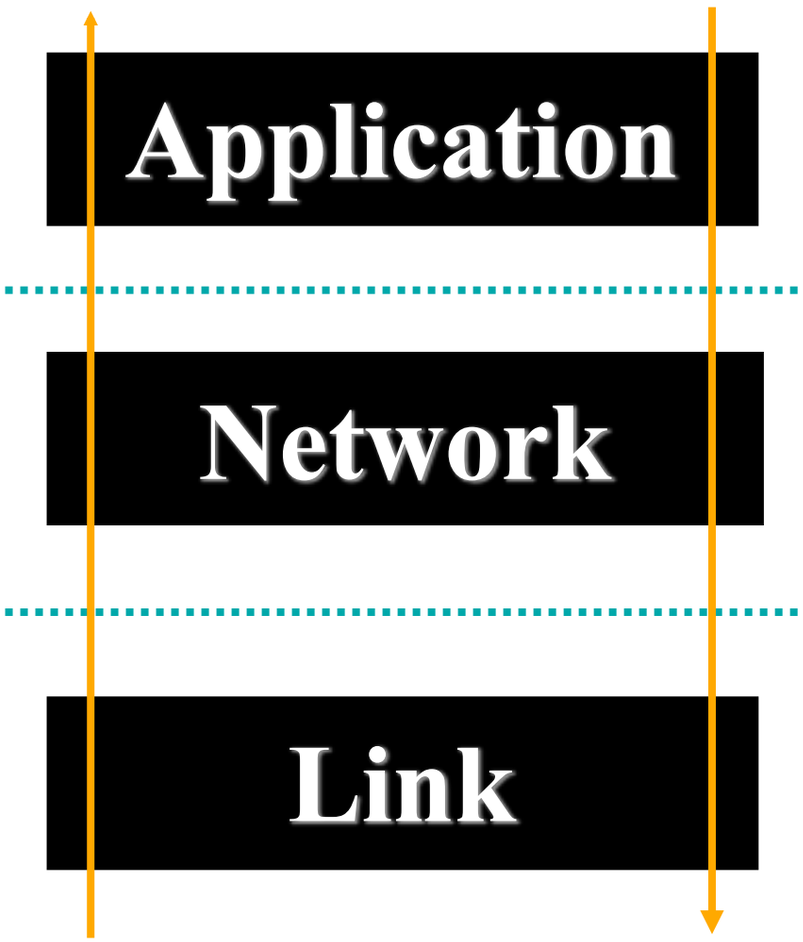
- All network flows were possible
  - Into or out of our network
  - To/from individual hosts and their processes
  - We need to control access to protect confidentiality, integrity and secrecy
    - What mechanism do we need?

Firewalls

- A firewall ... is a physical barrier inside a building or vehicle, designed to limit the spread of fire, heat and structural collapse.

Filtering: Firewalls

- Filtering traffic based on *policy*
  - Policy determines what is acceptable traffic
  - Access control over traffic
  - Accept or deny
- May perform other duties
  - Logging (forensics, SLA)
  - Flagging (intrusion detection)
  - QoS (differentiated services)



X-Listing

- **Blacklisting** - specifying specific connectivity that is explicitly disallowed
  - E.g., prevent connections from badguys.com
- **Whitelisting** - specifying specific connectivity that explicitly allowed
  - E.g., allow connections from goodguys.com

- These is useful for IP filtering, SPAM mitigation, …
- Q: What access control policies do these represent?

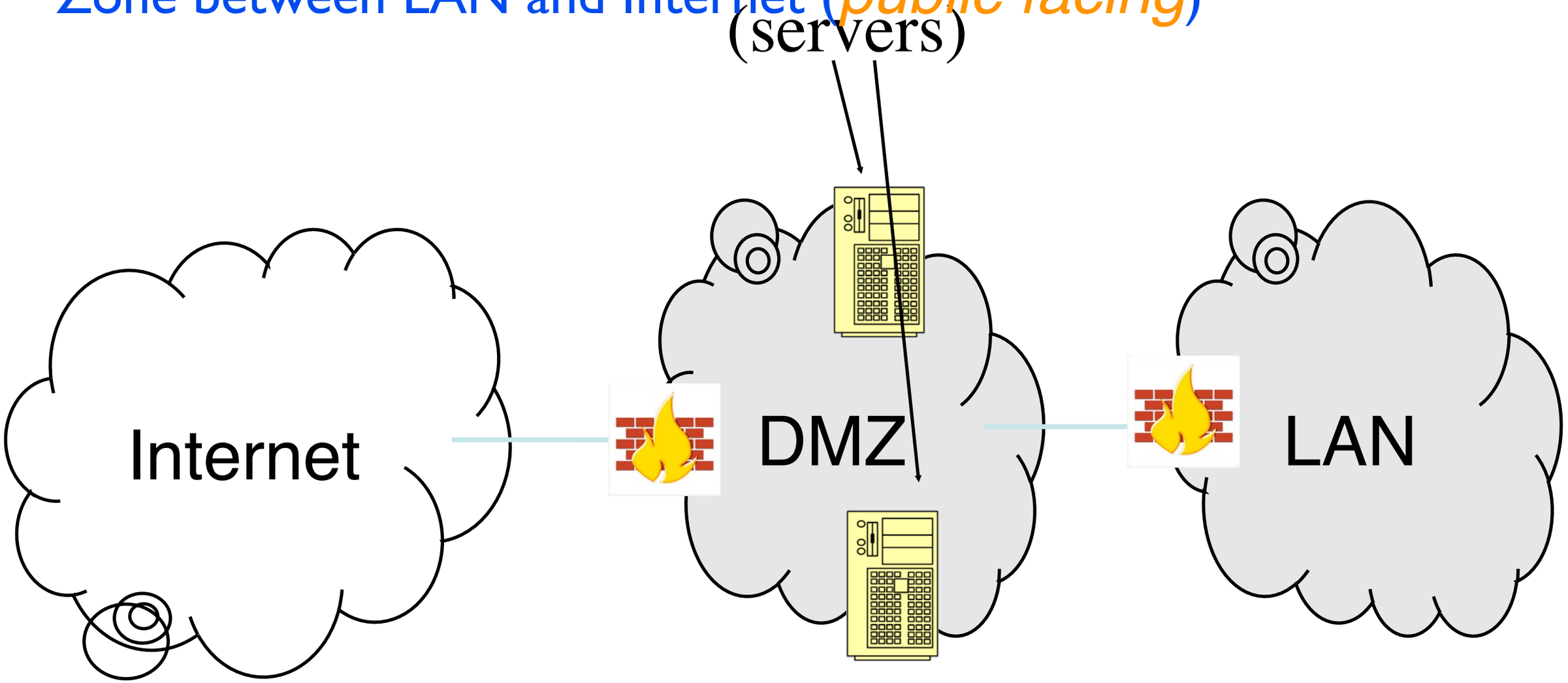
Stateful, Proxy, and Transparent

PennState

- Single packet may not contain sufficient data to make access control decision
  - Stateful: allows historical context consideration
  - Firewall collects data over time
    - e.g., TCP packet is part of established session
- Firewalls can affect network traffic
  - Transparent: appear as a single router (network)
  - Proxy: receives, interprets, and reinitiates communication (application)
  - Transparent good for speed (routers), proxies good for complex state (applications)

DMZ (De-militarized Zone)

- Zone between LAN and Internet (*public facing*)



Practical Issues and Limitations

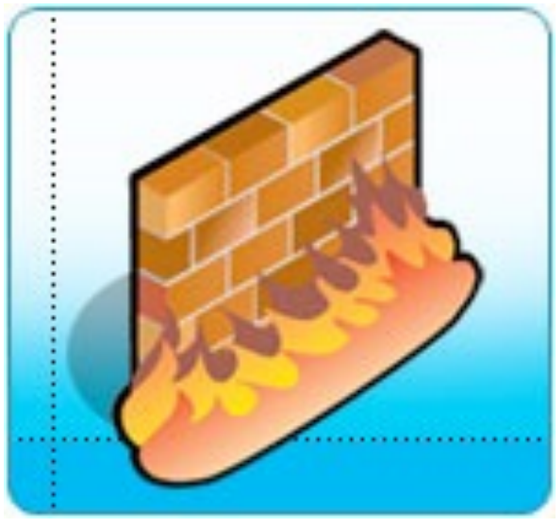
- Network layer firewalls are first defense
  - DMZs allow multi-tiered fire-walling
  - Tools are widely available and mature
  - Depth: application and personal firewalls
- Issues
  - Network perimeters not quite as clear as before
    - E.g., telecommuters, VPNs, wireless, …
  - Every access point must be protected
    - E.g., this is why war-dialing/driving is effective
  - Hard to debug, maintain consistency and correctness
  - Often seen by non-security personnel as impediment
    - E.g., Just open port *X* so I can use my wonder widget …

IP Firewall Policy

- Specifies what traffic is (not) allowed
  - Maps attributes to address and ports
  - Example: HTTP should be allowed to any external host, but inbound only to web-server

Source		Destination		Protocol	Flags	Actions
Address	Port	Address	Port			
*	*	1.1.1.1	80	TCP	SYN	Accept
1.1.1.*	*	*	80	TCP	SYN	Accept
*	*	*	80	TCP		Accept
*	*	*	*	TCP		Deny

Practical Firewall Implementations

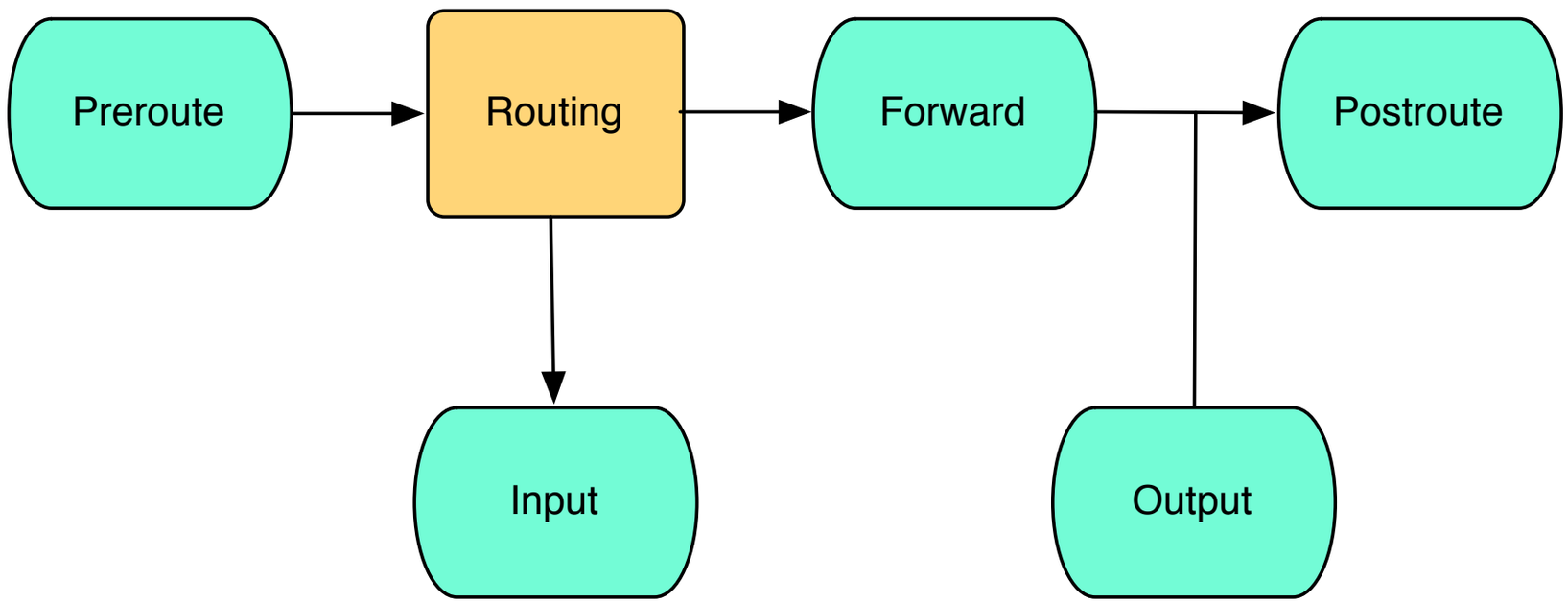


- Primary task is to filter packets
  - But systems and requirements are complex
- Consider
  - All the protocols and services
  - Stateless vs. stateful firewalls
  - Network function: NAT, forwarding, etc.

- Practical implementation: Linux iptables
  - http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.html
  - http://linux.web.cern.ch/linux/scientific3/docs/rhel-rg-en-3/ch-iptables.html

Netfilter hook

- Series of hooks in Linux network protocol stack
- An `iptable` rule set is evaluated at each
  - "PREROUTING": anything received
  - "INPUT": inbound to local destination
  - "FORWARD": inbound/outbound but routed off host
  - "OUTPUT": outbound to remote destination
  - "POSTROUTING": anything outbound



iptables Concepts

- *Table*: all the firewall rules
- *Chain*: list of rules associated with the chain identifier, e.g., hook name (like a subroutine)
- *Match*: when all a rule's field match the packet
- *Target*: operation to execute on a packet given a match

The iptables firewall looks in the firewall *table* see if a rule in the *chain* associated with the current hook *matches* a packet, and executes the rule's *target* if it does.

Test it out

- PING on localhost
  - *ping -c 1 127.0.0.1*
- Add iptables rule to block
  - *iptables -A INPUT -d 127.0.0.1 -p icmp -j DROP*
- Try ping
- Delete the rule
  - *iptables -D INPUT 1*
  - *iptables -D INPUT -d 127.0.0.1 -p icmp -j DROP*
  - *iptables -F INPUT*

Testing

- Use loopback to test the rules locally on your machine
  - IP address 127.0.0.1
- ICMP
  - submit ping requests to 127.0.0.1 as above
- TCP
  - submit requests to 127.0.0.1 at specific port
    - server
      - nc -l -p 3750
      - listen at port 3750
    - client
      - nc -p 3000 localhost 3750
      - send from port 3000 to localhost at port 3750

Deep Packet Inspection

PennState

- *Deep packet inspection* looks into the internals of a packet to look for some application/content context
  - e.g., inspect HTTP for URLs that point to malicious websites
  - Can have serious privacy issues if done by, say COMCAST

- To specify a match in `iptables`
  - iptables -A INPUT -p tcp -m string --algo bm --string 'exe'
    - matches to packet with content containing 'exe'
  - iptables -A INPUT -p tcp -m length --length 10:100
    - matches to packet with length between 10 and 100 bytes
    - Also, can specify 'greater than 10' by 10:

Firewall Policy Design

- So, what is the problem with the firewall rules...

```
accept tcp 192.168.0.0/16 any
deny tcp 192.168.1.0/24 any 3127
```

- This may be a simple problem, but
- Rules now have complex actions

FIREMAN

- Static analysis tool for detecting incorrect, inefficient, or inconsistent firewall rules
  - Using something called binary decision diagrams
- Finds real misconfigurations
  - Classify misconfigurations
  - Applies intra- and inter-firewalls

Misconfigurations

- Consider the following rules

1.	deny tcp 10.1.1.0/25 any
2.	accept udp any 192.168.1.0/24
3.	deny tcp 10.1.1.128/25 any
4.	deny udp 172.16.1.0/24 192.168.1.0/24
5.	accept tcp 10.1.1.0/24 any
6.	deny udp 10.1.1.0/24 192.168.0.0/16
7.	accept udp 172.16.1.0/24 any

- Compare Rules 2 and 4
- Compare Rules 1, 3, and 5
- Compare Rules 4 and 7
- Compare Rules 2 and 6

Misconfigurations

- Violations
  - What is the security goal?
- Inconsistencies (possibly between firewalls)
  - Shadowing: Accept (denies) *all* packets already denied (accepted) - E.g., 2 and 4
  - Generalization: Excluded a subset of preceding - E.g., 4 and 7
  - Correlation: Matches subset of preceding, but takes a different action - E.g., 2 and 6
- Inefficiencies
  - Redundancy: Remove rule and no change
  - Verbosity: Summarize with fewer rules

Analysis

- What is static analysis?
  - Analyze without running program (firewall rules)
  - Approximate all possible executions at once
- For a firewall
  - Track all packets that have been accepted (A), denied (D), diverted (F) before this rule - remaining (R) is implied
  - jth rule defines $<P_j, action_j>$
  - A_j, D_j, F_j identify the packets accepted, denied, or diverted prior to rule j
- Analysis
  - Update the state of A, D, F, R at each rule
  - Evaluate for shadowing, generalization, correlation, etc.

Analysis Rules

- Problems detected by comparing sets (A, D, F, R, P)
  - In a good rule, packets affected are only in remaining
  - For an bad deny rule, suppose P_j and R_j have no intersection (always a problem)
    - (P_j, Deny) where P_j subset A_j - shadowing
      - Already accepted all the packets to be denied here
    - (P_j, Deny) where (P_j intersect R_j) = NULL and (Pj intersect Aj) = NULL - redundant
      - Already denied remaining and wouldn't block accepted
  - For a maybe bad deny rule, if P_j and R_j are not related by subset and only related by a partial intersection
    - P_i and D_i have an intersection - correlation

Analysis Example

- Consider the following rules

1.	deny tcp 10.1.1.0/25 any
2.	accept udp any 192.168.1.0/24
3.	deny tcp 10.1.1.128/25 any
4.	deny udp 172.16.1.0/24 192.168.1.0/24
5.	accept tcp 10.1.1.0/24 any
6.	deny udp 10.1.1.0/24 192.168.0.0/16
7.	accept udp 172.16.1.0/24 any

- Rules for A: 2, 5, 7 — Rules for D: 1, 3, 4, 6
- At Rule 4: P_4 has no intersection with remaining R_4
  - any ›192.168.1.0/24 in A_4 (from Rule 2)
  - P_4 is a subset of A_4 — Shadowing
- At Rule 6:
  - Traffic in P_6 intersects of A_6 (from Rule 2) — Correlation

Take Away

- A firewall is an authorization mechanism for network flows
  - Control packet flows to subnets, hosts, ports
  - Scan a rulebase for matching rule for packet
    - Like Windows ACLs, but with default accept
- We examined the Linux iptables firewall
  - Netfilter hooks provide complete mediation
  - Rule chains can be connected like subroutines
- However, firewall rules may be misconfigured
  - FIREMAN detects violations, inconsistencies, and inefficiencies using static analysis of rule bases
    - Compare sets of packets at rule with those accepted, denied, etc.