



PennState

CSE 543: Computer Security

Module: Firewalls

Prof. Syed Rafiul Hussain
Department of Computer Science and Engineering
The Pennsylvania State University

- **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?

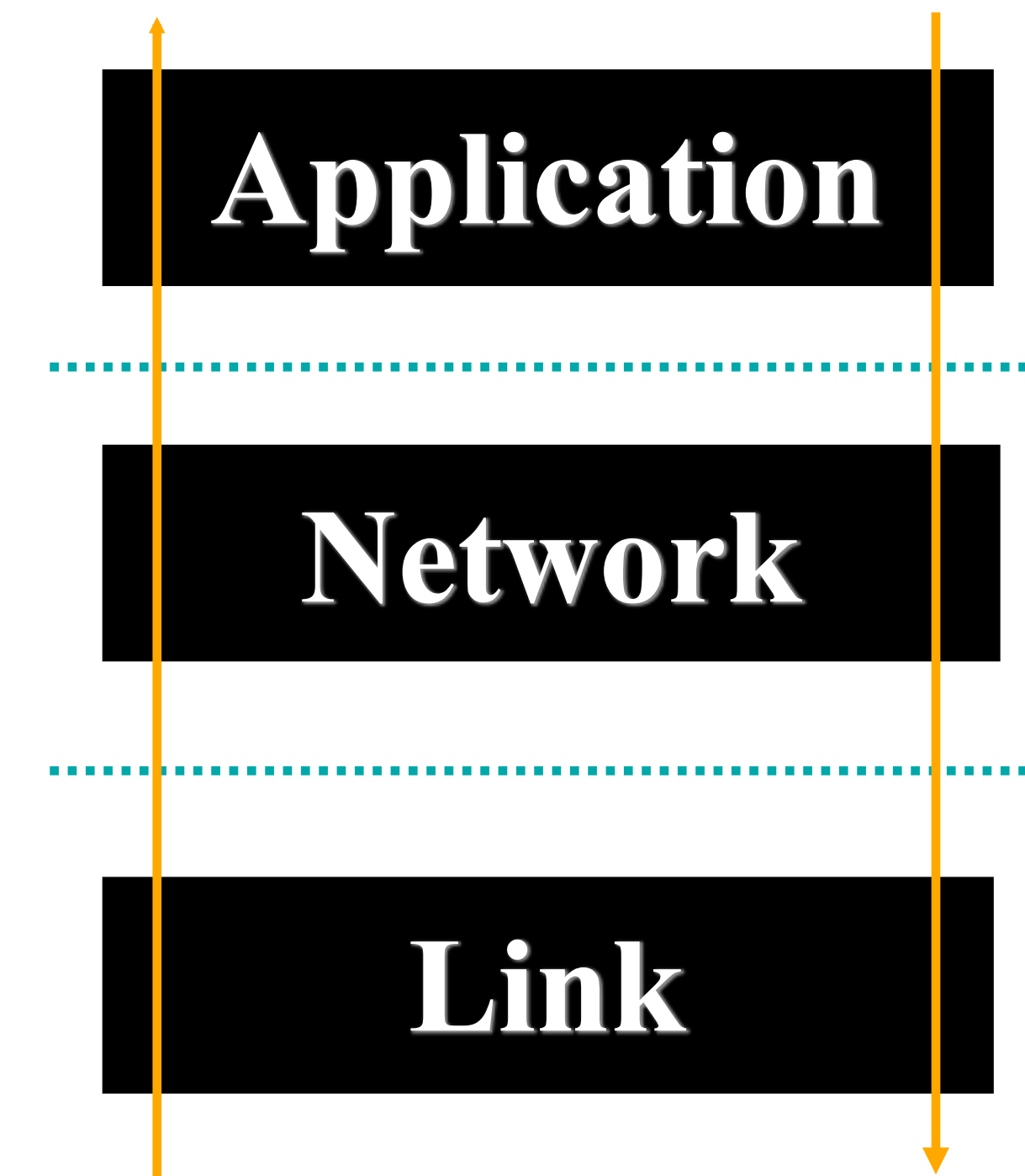


- A firewall ... is a physical barrier inside a building or vehicle, designed to limit the spread of fire, heat and structural collapse.



Filtering: Firewalls

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



- **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

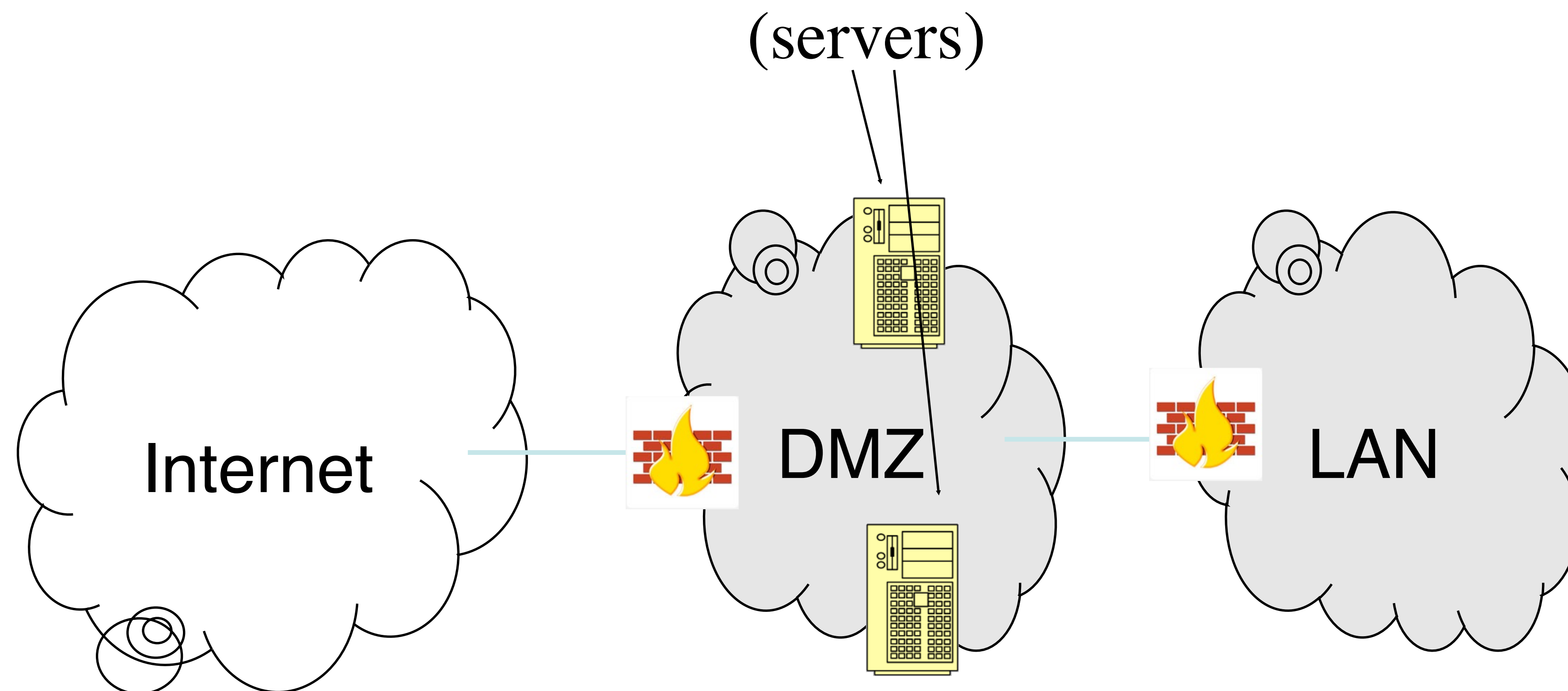
- These is useful for IP filtering, SPAM mitigation, ...
- Q:What access control policies do these represent?



- 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
 - ▶ **Routed**: appear as a single router (network)
 - ▶ **Proxy (Circuit-level 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*)

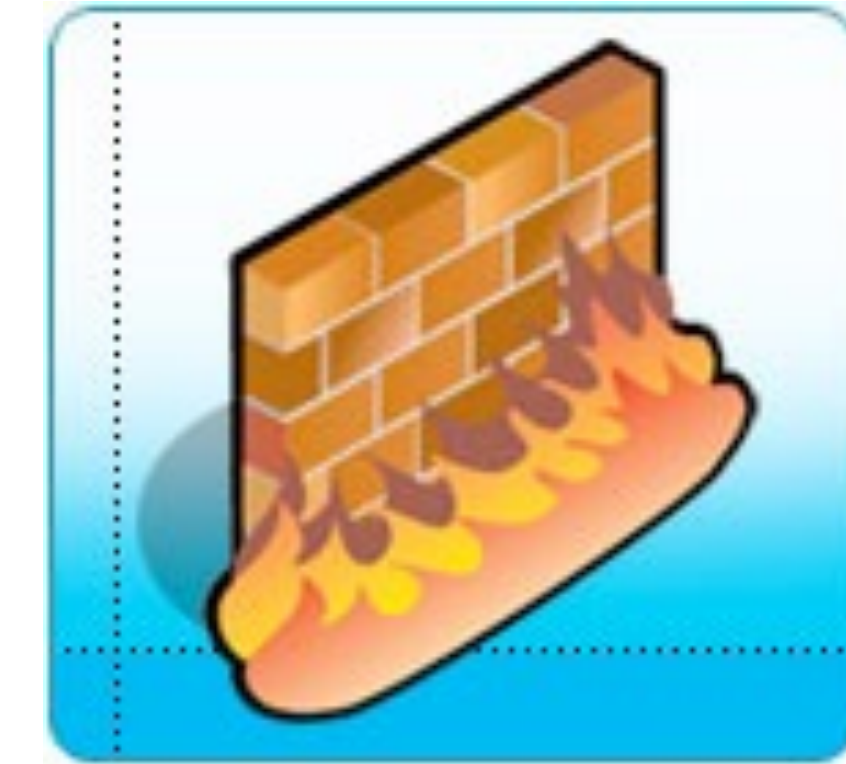


IP Firewall Policy

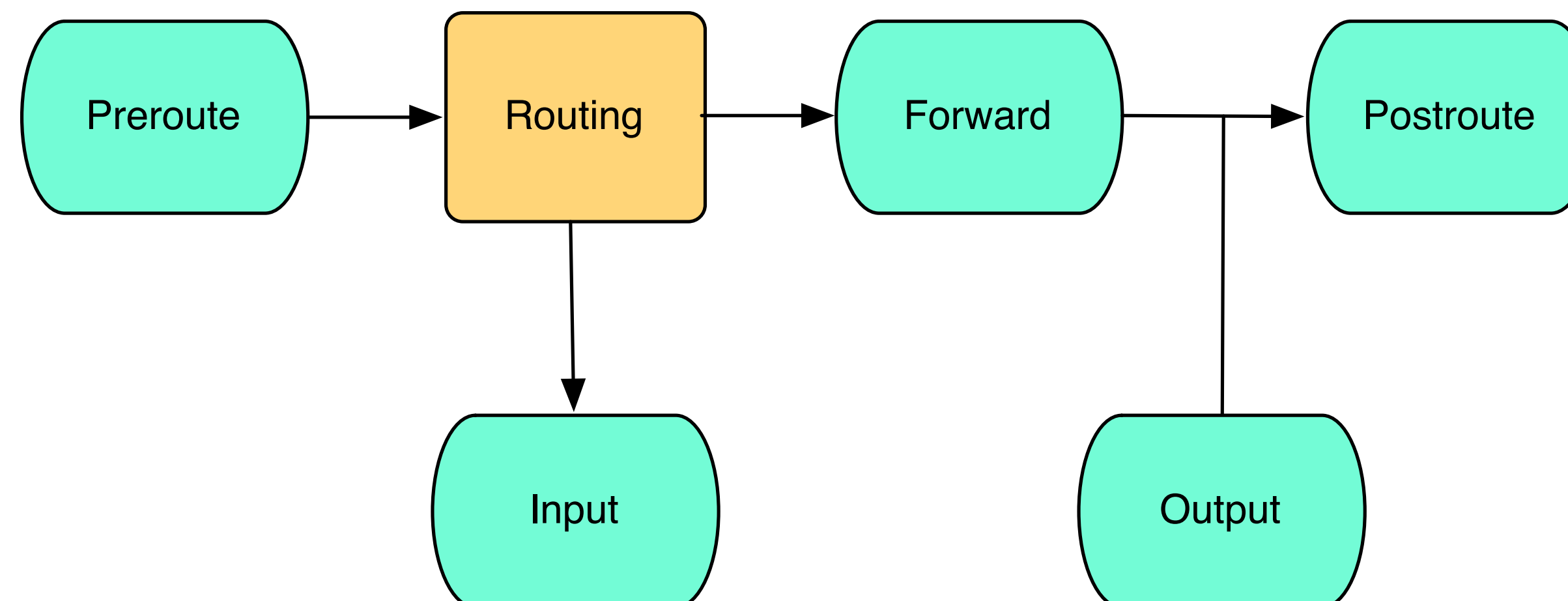
- Specifies what traffic is (not) allowed
 - ▶ Maps attributes to address and ports
 - ▶ Example: HTTP should be allowed to any external host, but inbound only to web-server

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

- **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>



- Series of hooks in Linux network protocol stack
- An `iptables` 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



- **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** to seek if a rule in the **chain** associated with the current hook **matches** a packet, and executes the rule's **target** if it does.

- 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*

Deep Packet Inspection

- *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



- 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

- 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

- 2 accepts all packets that would be denied by 4 (shadowing)
- 1 and 3 deny all packets that would be accepted by 5 (shadowing)
- subnet mask: /n where n is the number of bits retained in the IP address, so address of 5 includes 3 and 1
- 4 denies subset of connections accepted by 7 (generalization)
- 2 accepts an intersecting set of connections denied by 6 (correlation)
- generalization and correlation may be OK

- **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

- 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
 - ▶ j th rule defines $\langle P_j, \text{action}_j \rangle$
 - ▶ 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 (Fireman)

- 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 $(P_j \cap R_j) = \text{NULL}$ and $(P_j \cap A_j) = \text{NULL}$ - **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_j and D_j 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 \supset 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**

- 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.