Network Endpoint Security

Firewalls, IDS, and IPS

Content

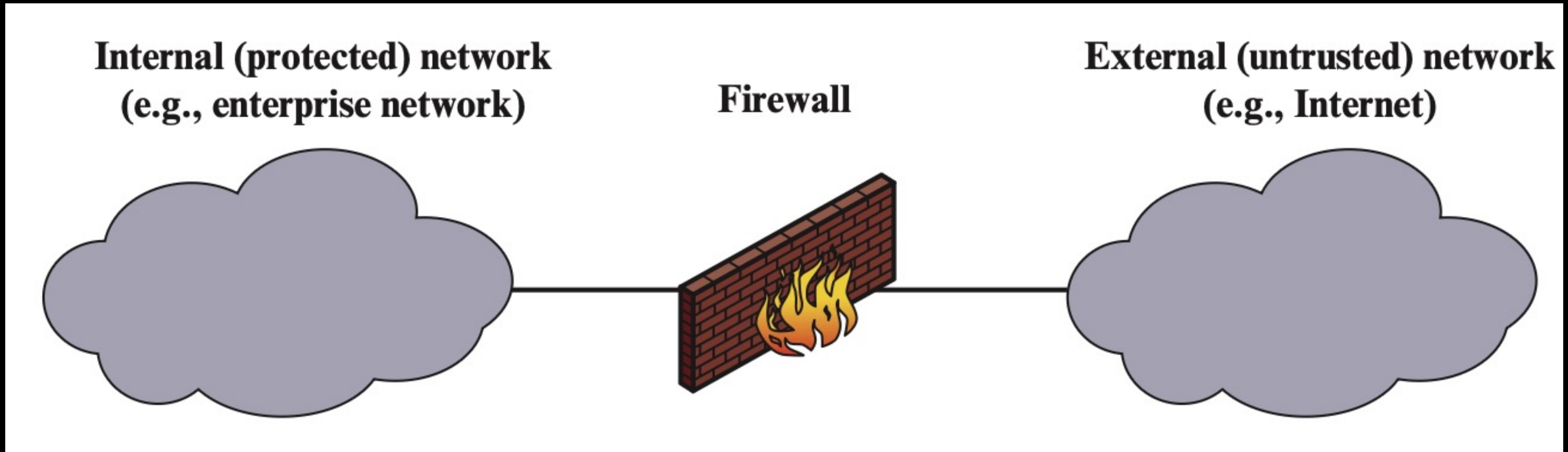
	Firewalls, IDS, IPS
➡	Firewalls
	Intrusion Detection Systems
	Intrusion Prevention Systems
	Linux iptables
	Linux iptables: Experimenting with Stateless Firewall Rules
	Linux iptables: Connection Tracking and Stateful Firewall

Firewalls

- A protection service inserted between your network and the internet
- They can be hardware or software

Firewalls

- Protects the premises network from internet attacks
- They can be deployed internal to the enterprise network to segregate portions of the network


Internal (protected) network
(e.g., enterprise network)
Firewall
External (untrusted) network
(e.g., Internet)

(a) General model

Firewalls

- Firewalls characteristics

All traffic from inside to outside, and vice versa, must pass through the firewall.
Only authorized traffic, as defined by the local security policy, will be allowed to pass.
The firewall itself is immune to penetration. This implies the use of a hardened system with a secured operating system.

Firewalls

- What firewalls can do

Service control	• Determine the types of Internet services that can be accessed, inbound or outbound • Filter the traffic based on the IP address, port number, or protocol
Direction control	• The direction of the service: inbound or outbound o Inbound → from outside to my network o Outbound → from my network to outside
User control	• Who can and cannot access a particular service • Applied to users inside the firewall perimeter • Can be applied external users; requires secure authentication (e.g., IPSec)
Behavior control	• Control how a particular service is used • E.g., filter emails to eliminate spams

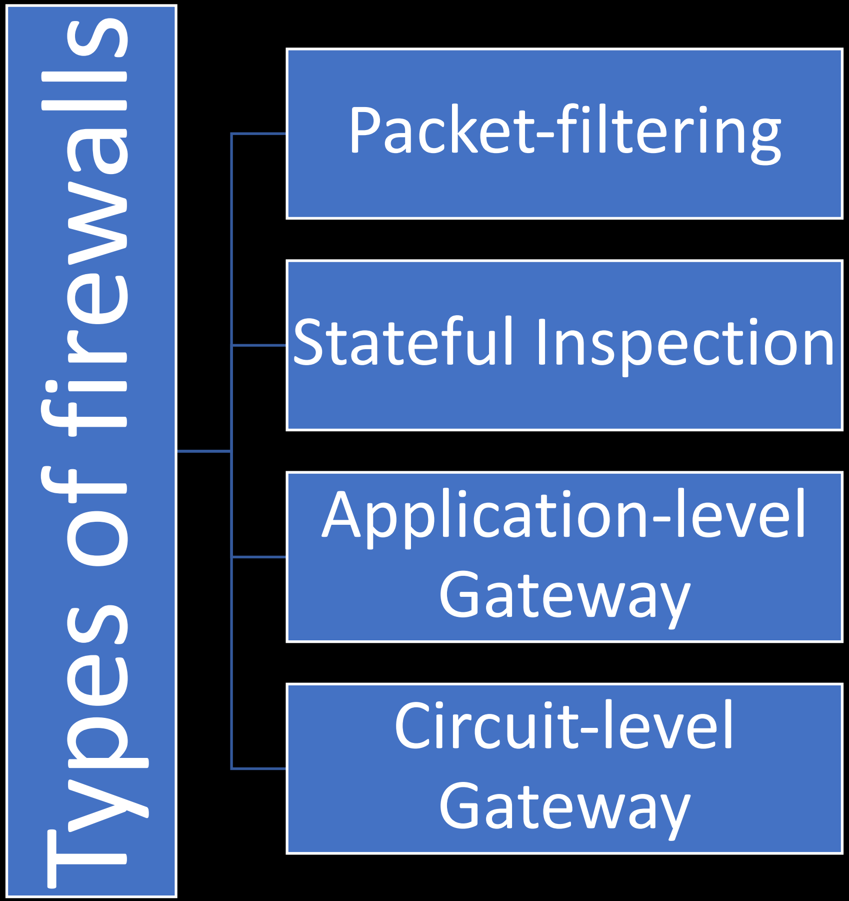
Firewalls

- What firewalls **cannot** do

Protect against attacks that bypass the firewall
Protect against internal threats
Protect against an improperly secured wireless LAN
Protect against an infected device connected to the internal network

Firewalls

- Types of firewalls



Types of firewalls

Packet-filtering

Stateful Inspection

Application-level Gateway

Circuit-level Gateway

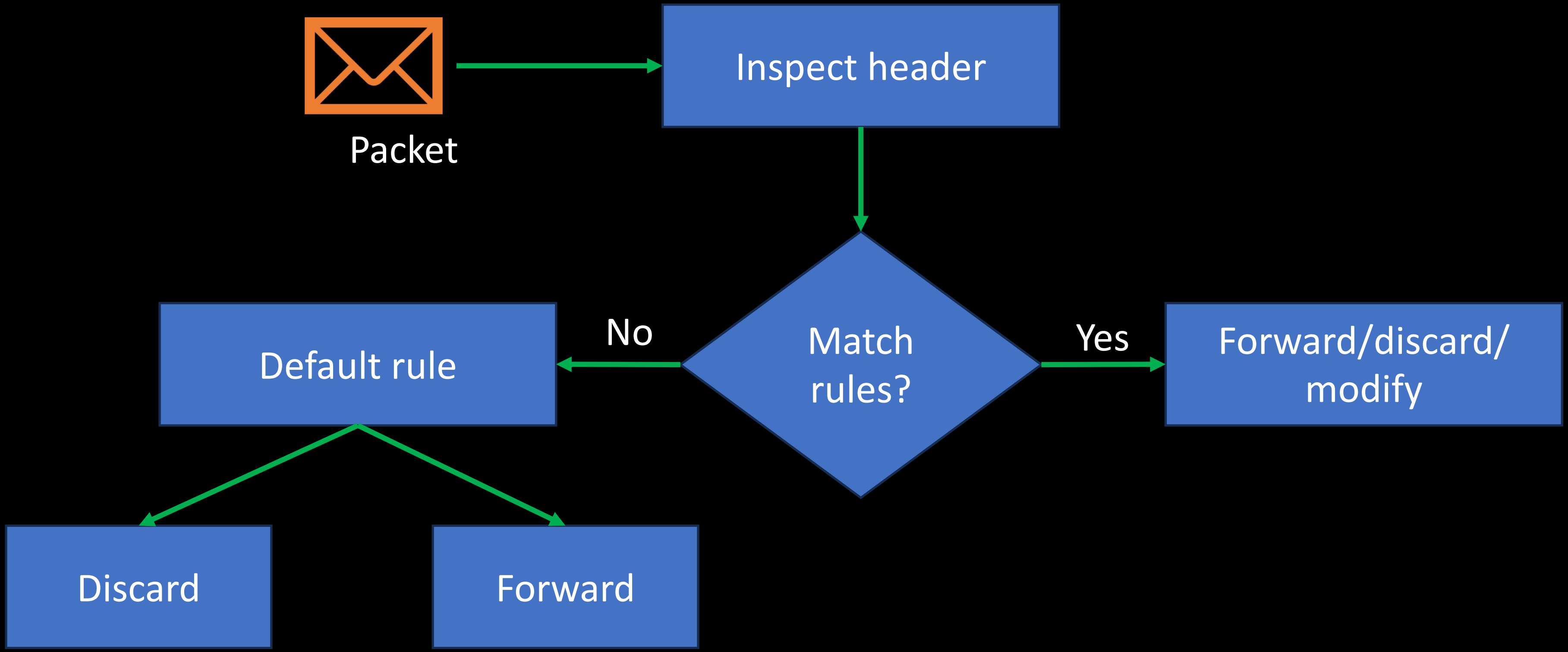
Firewalls
Packet-filtering Firewall

- Applies a set of rules to each incoming and outgoing IP packet
- It either forwards the packet or discards it
- Filtering rules are based on information contained in a network packet:

Source IP address
Destination IP address
Source and destination transport-level address. This refers to the port number on which specific applications working on (e.g., TELNET or HTTP)
IP protocol field
Interface – it refers to which interface of the firewall the packet came from or which interface of the firewall the packet is destined for

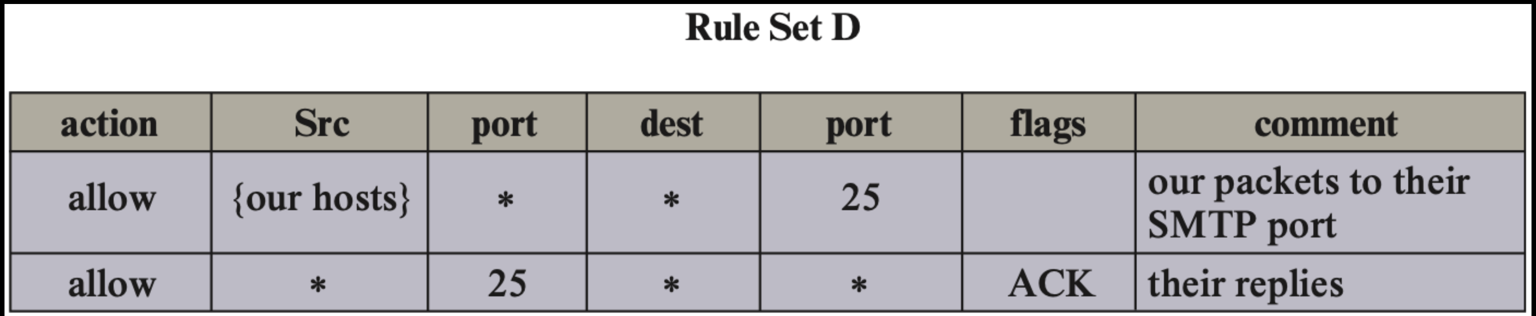
Firewalls
Packet-filtering Firewall

- When a packet arrives at the firewall



Firewalls
Packet-filtering Firewall

- Example: The first rule allows outgoing SMTP traffic from your hosts to any destination on port 25, enabling your systems to send emails to external SMTP servers.
- The second rule allows incoming traffic from any source on port 25 with the ACK flag, permitting responses (replies) from external SMTP servers back to your hosts, completing the two-way communication for email exchange.

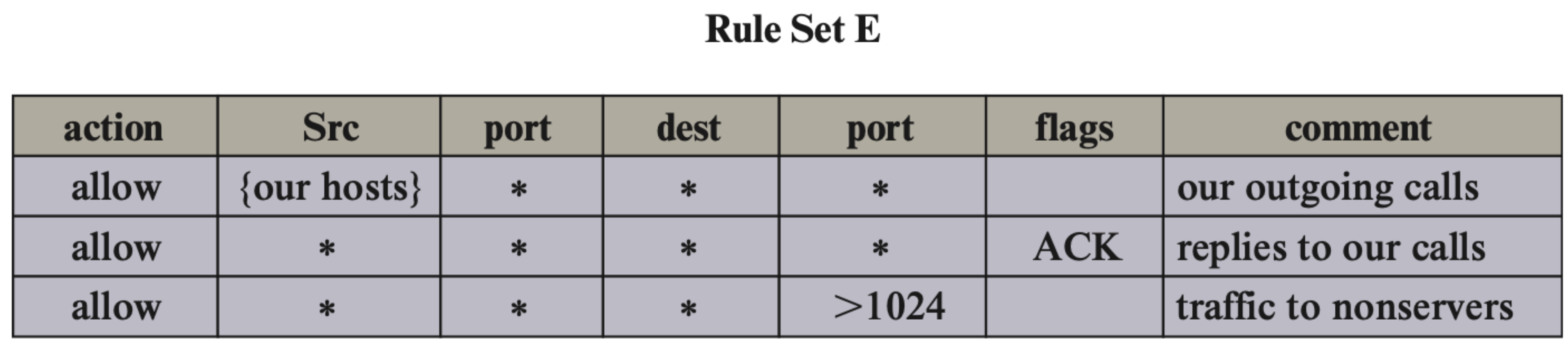


Rule Set D

action	Src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

Firewalls
Packet-filtering Firewall

- Example: this rule set allows
  - Packets that originate internally
  - Reply packets to a connection initiated by an internal machine
  - Packets destined for a high-numbered port on an internal machine



Rule Set E

action	Src	port	dest	port	flags	comment
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

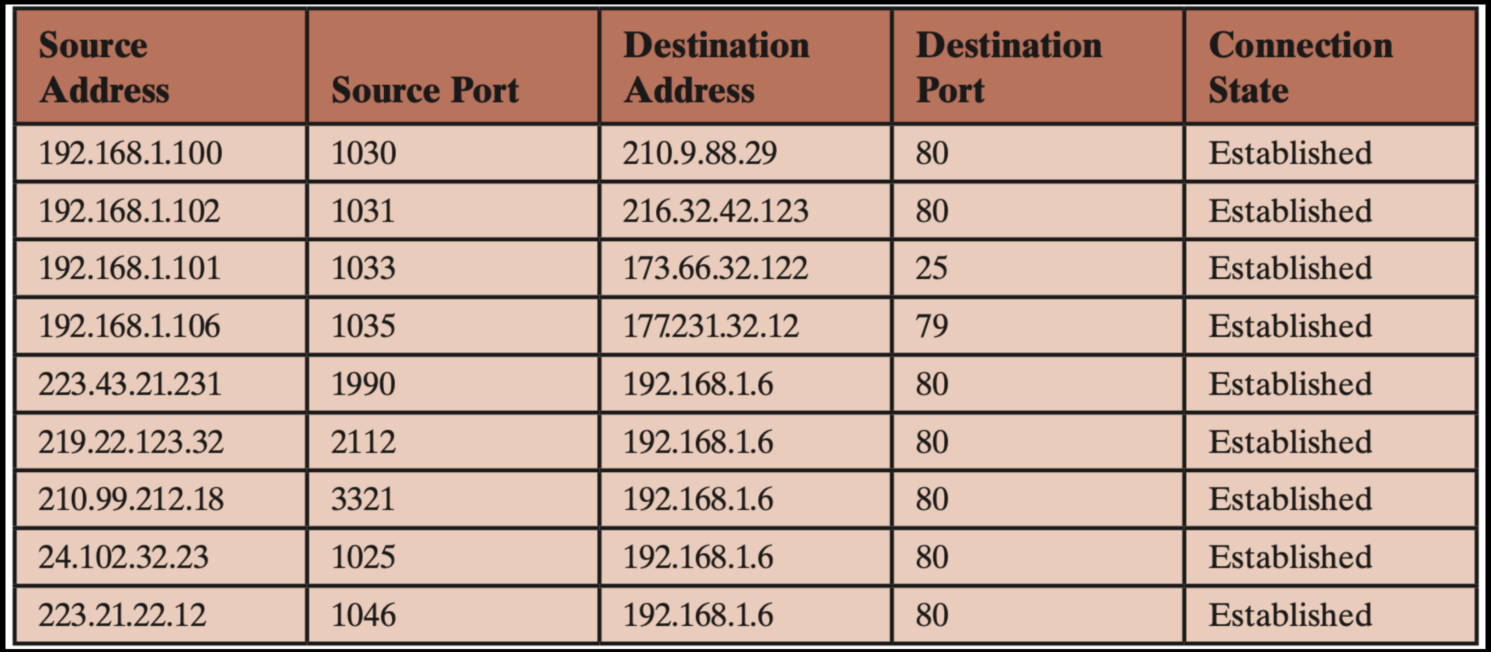
Firewalls
Packet-filtering Firewall

- Some attacks on packet-filtering firewalls

Attack	Description	Countermeasure
IP address spoofing	Change the IP address to one of the addresses of an internal host	Discard packets with an inside source address if the packet arrives on an external interface
Source routing attacks	Specify the route of the packets to reach the destination	Discard all packets that use this option
Tiny fragment attacks	Create small fragments of the packet and force the TCP header information into a separate packet fragment	Enforce a minimum packet size to include a predefined minimum amount of the transport header

Firewalls
Stateful Inspection Firewall

- Keeps track and monitors the state of active network connections
- Inspects individual packets as a packet filtering firewall + records information about connections



Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.100	1030	210.9.88.29	80	Established
192.168.1.102	1031	216.32.42.123	80	Established
192.168.1.101	1033	173.66.32.122	25	Established
192.168.1.106	1035	177.231.32.12	79	Established
223.43.21.231	1990	192.168.1.6	80	Established
219.22.123.32	2112	192.168.1.6	80	Established
210.99.212.18	3321	192.168.1.6	80	Established
24.102.32.23	1025	192.168.1.6	80	Established
223.21.22.12	1046	192.168.1.6	80	Established

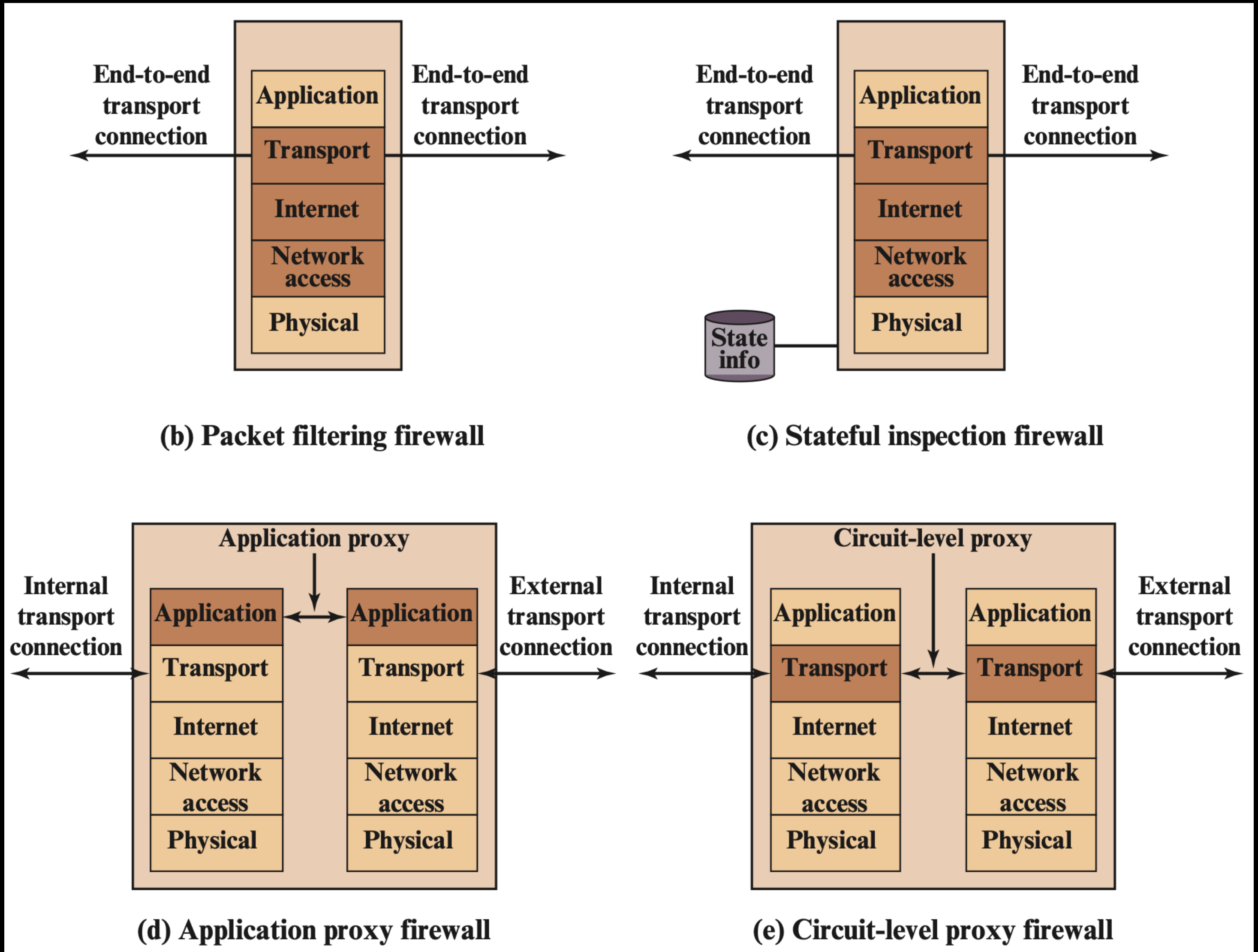
Firewalls
Application-level Gateway

- Also called **application proxy**
- Governs traffic to, from, or by an application or service
  - E.g., Telnet and FTP
- More secure than packet filters:
  - Rather than combining rules on the TCP/IP level, it needs only scrutinize a few allowable applications
  - Easy to log and audit all incoming traffic at the application level
- Disadvantage: additional processing overhead on each connection

Firewalls
Circuit-level Gateway

- Also called **circuit-level proxy**
- A stand-alone system provides connection security to internal and external computers in a network's session layer
- Verifies the TCP/UDP packets on a virtual circuit between two transport layers
  - It does not inspect the contents of the packets
- If the packet contains invalid header information or breaches other firewall rules, the traffic is blocked, and the connection is terminated

Firewalls



(b) Packet filtering firewall

(c) Stateful inspection firewall

(d) Application proxy firewall

(e) Circuit-level proxy firewall

Firewalls

Firewall type	OSI layer	How it works	Pros	Cons
Packet-filtering	Network and Transport layers (L3, L4)	Examines individual packets based on source/destination IP addresses, ports, and protocols	• Fast and efficient • Simple to implement	• Doesn't track packet state or connections • Vulnerable to spoofing and some types of attacks
Stateful Inspection	Network and Transport layers (L3, L4)	Tracks active connections and ensures packets are part of a valid session	• Monitors connection state and context	
Application-Level (application proxy)	Application Layer (L7)	Acts as a proxy, examining and filtering traffic at the application level (e.g., HTTP, FTP)	• Deep packet inspection • Blocks specific content, commands, or user actions	• Slower due to deep inspection • Requires more resources
Circuit-Level (circuit proxy)	Session Layer (L5)	Verifies that TCP/UDP sessions are valid; doesn't inspect the content	• Efficient for session management • Hides internal network structure	• No application-level filtering • Limited control over content

Content

	Firewalls, IDS, IPS
	Firewalls
➡	Intrusion Detection Systems
	Intrusion Prevention Systems
	Linux iptables
	Linux iptables: Experimenting with Stateless Firewall Rules
	Linux iptables: Connection Tracking and Stateful Firewall

Intrusion Detection Systems

Intrusion

- Violations of security policy, affect the confidentiality, integrity, or availability of a computer or network.

Intrusion detection

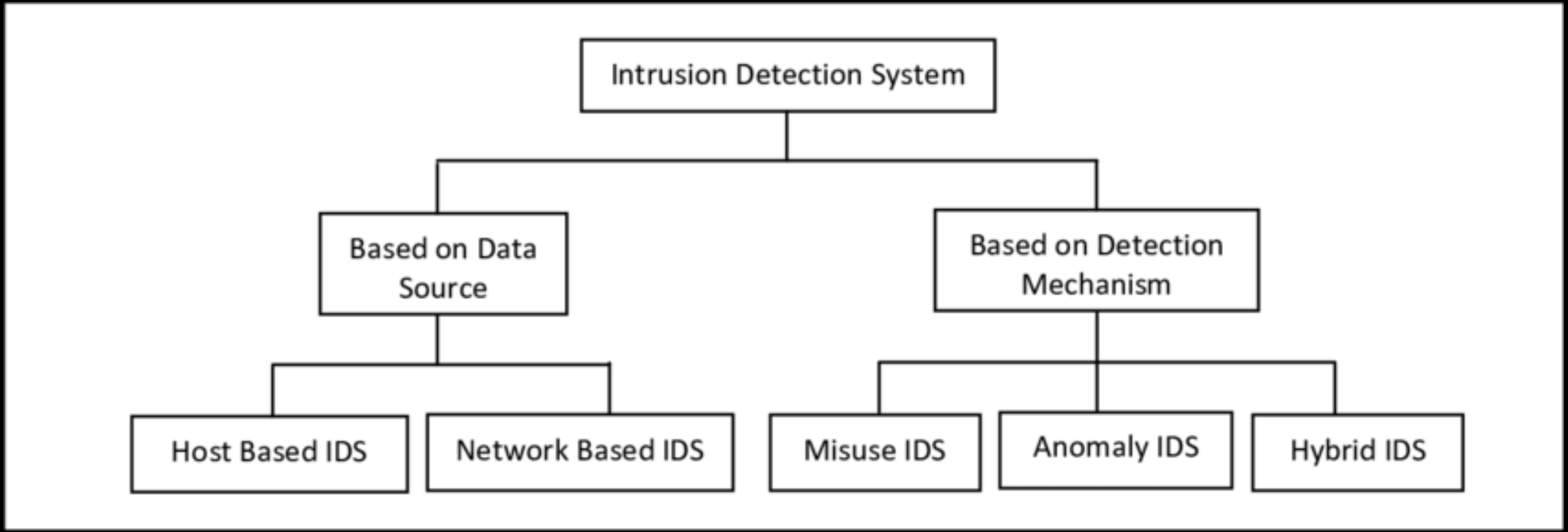
- The process of collecting information about events occurring a computer system or network and analyzing them for signs of intrusions.

Intrusion detection system

- HW/SW products that gather and analyze information within a computer or a network to find and provide real-time warning of attempts to access a system in an unauthorized manner.

Intrusion Detection Systems

- Classification of IDSs



Intrusion Detection Systems

Host-based IDS

- Monitors a single host and the events occurring within that it for suspicious activity
- Determine exactly which processes and user accounts are involved in the attack
- Can more readily see the intended outcome of an attempted attack
    - They access and monitor the data files and system processes usually targeted by attacks

Network-based IDS

- Monitors network traffic for particular network segments or devices
- Analyzes network, transport, and application protocols to identify suspicious activity

Intrusion Detection Systems

- An IDS comprises three logical components

1. **Sensors**:
   - Collect data and pass it to the analyzer
   - The input can be packets, system call traces, log files, etc.
2. **Analyzer**:
   - Determines if an intrusion has occurred
   - Outputs evidence supporting the conclusion that intrusion has occurred + guidance on what actions to take
3. **User interface**:
   - Enables a user to view output from the system or control the behavior of the system

Intrusion Detection Systems

- IDS can be classified based on the detection mechanism:

Misuse detection	**Anomaly detection**
• Uses pattern-matching algorithms • Uses DBs of attack patterns, or signatures • Its rules specify system events that can be symptomatic of security incident	• Searches for activity that is different from the normal behavior of system
Pros: accurate and generate few false alarms	**Pros**: detect unknown attacks
Cons: cannot detect novel or unknown attacks	**Cons**: trade-off between false positives and false negatives

Intrusion Detection Systems

- IDS can be classified based on the detection mechanism:

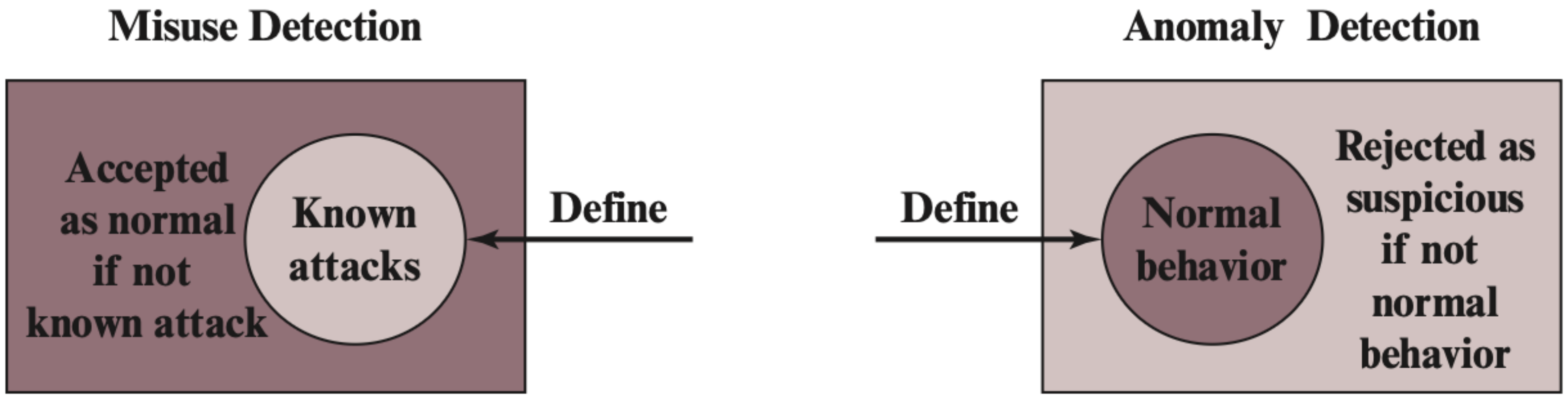
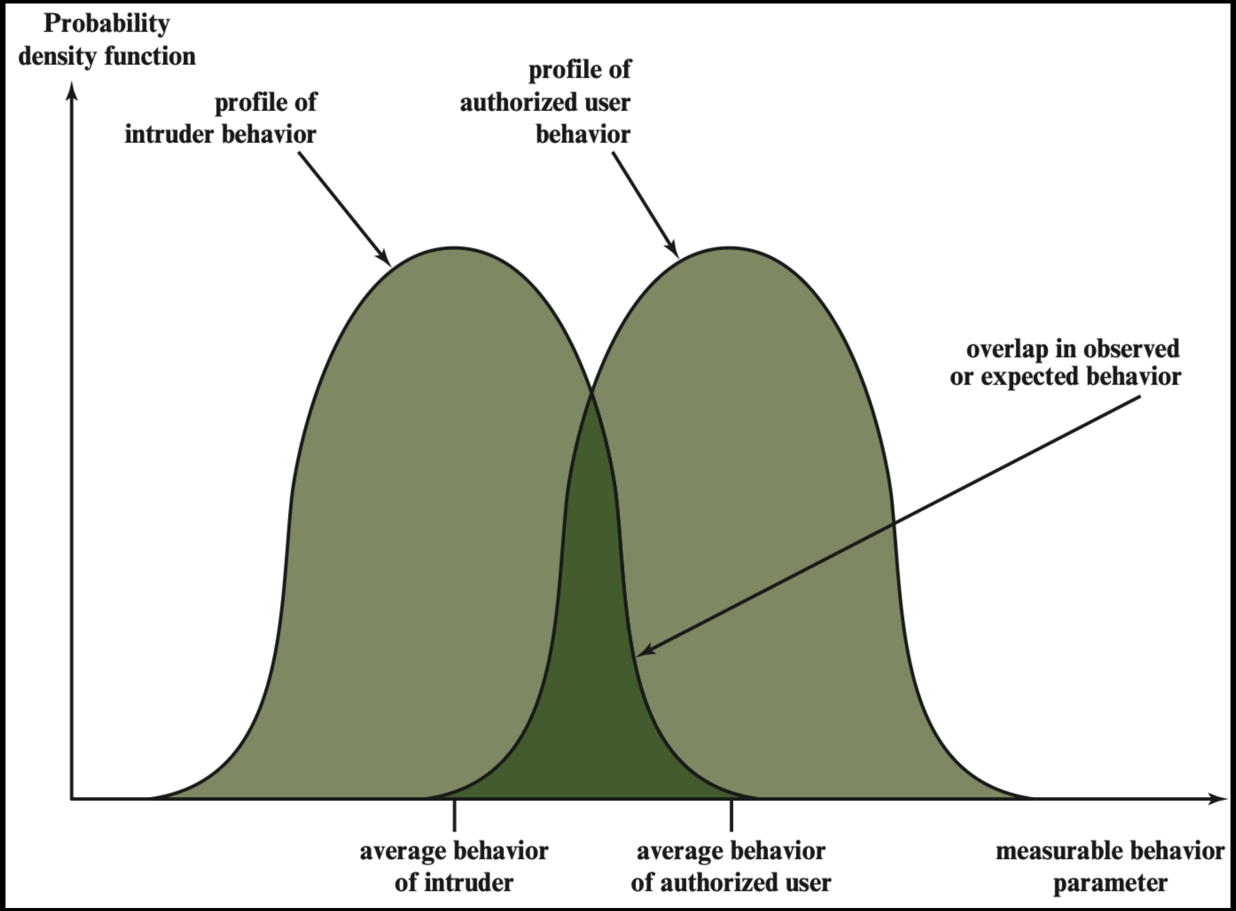

Misuse Detection
Accepted as normal if not known attack
Known attacks
Define
Define
Anomaly Detection
Normal behavior
Rejected as suspicious if not normal behavior

Figure 21.4 Approaches to Intrusion Detection

Intrusion Detection Systems

- Intruder behavior vs authorized user behavior



Intrusion Detection Systems
Host-Based Intrusion Detection Techniques

- HIDSs add a specialized layer of security software to vulnerable systems
    - e.g., database servers and administrative systems
- **Advantage**: it can detect both external and internal intrusions
    - this is not possible either with network-based IDSs or firewalls
- Use one or a combination of anomaly and misuse protection
- For anomaly detection, two common strategies are:
    - **Threshold detection**: Defining thresholds, independent of user, for the frequency of occurrence of various events
    - **Profile based**: Develop a profile of activity for each user, used to detect changes in the behavior of individual accounts

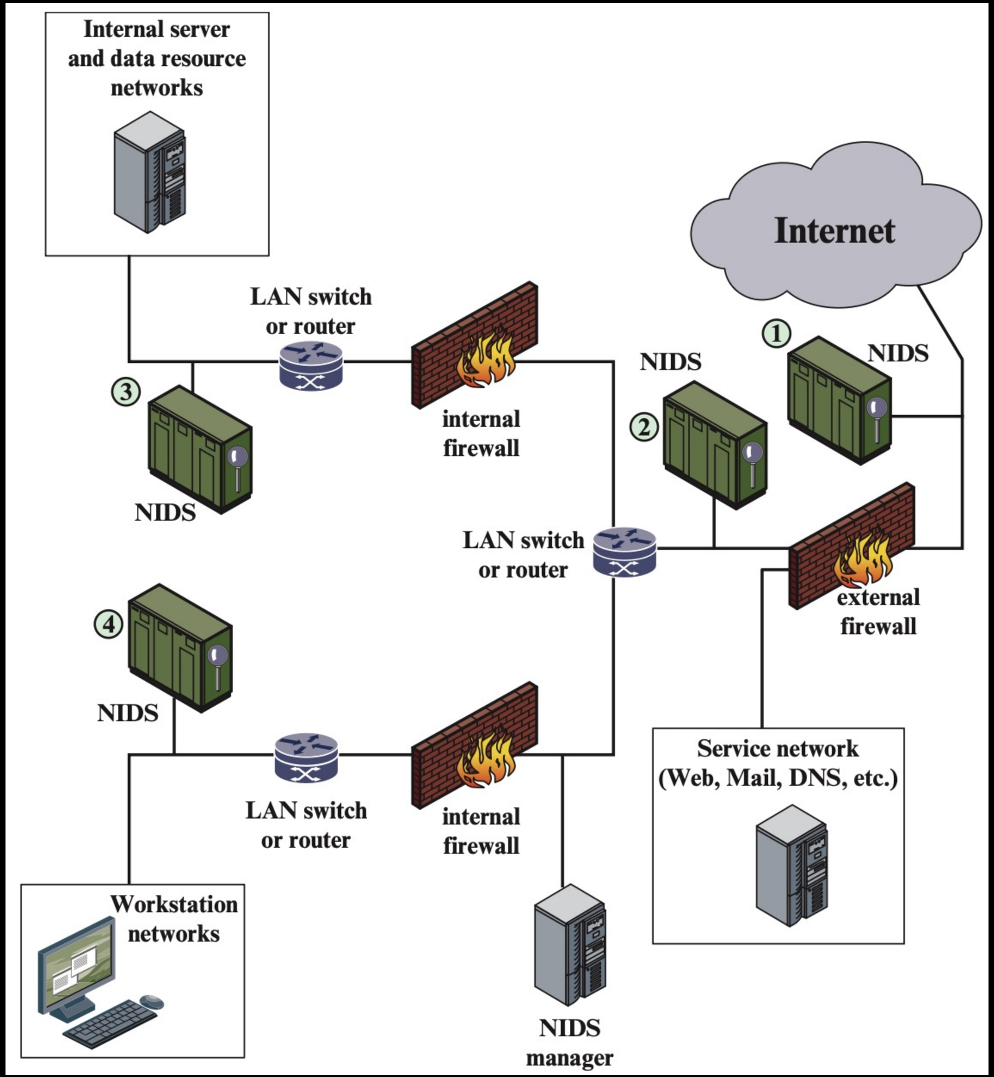
Intrusion Detection Systems
Network-Based Intrusion Detection Techniques

- Monitors the traffic on its network segment as a data source

- This is accomplished by placing the NIC in promiscuous mode
    - capture all network traffic that crosses its network segment

- Network traffic on other segments can’t be monitored by a single NIDS

Intrusion Detection Systems
Network-Based Intrusion Detection Techniques



Content

Firewalls, IDS, IPS
Firewalls
Intrusion Detection Systems
Intrusion Prevention Systems
Linux iptables
Linux iptables: Experimenting with Stateless Firewall Rules
Linux iptables: Connection Tracking and Stateful Firewall

Intrusion Prevention Systems

- IDS + capability to block detected malicious activity
  - a.k.a intrusion detection and prevention system (IDPS)
- It can be **host-based**, **network-based**, or **distributed/hybrid**
- It can use **anomaly detection** or **misuse detection**:
  - **Anomaly detection**: identify behavior that is not that of legitimate users
  - **Misuse detection**: use pattern matching or signatures

Intrusion Prevention Systems
Host-Based IPS

Examples of the types of malicious behavior addressed by a HIPS

Exploit	Description
Modifying system resources	rootkits, trojans, backdoors, and changing system directories, registry, libraries, and user accounts
Privilege-escalation exploits	Attempt to give ordinary users root access
Buffer overflow exploits	Software vulnerabilities
Access to e-mail contact list	Detect worms spread by copying themselves to addresses in the local system’s e-mail address book
Directory traversal	Allows the hacker to access files outside the range of what a server application user would normally need to access

Intrusion Prevention Systems
Host-Based IPS

- HIPS can use a sandbox approach
  - Quarantines a code in an isolated system area, then runs it and monitors its behavior
  - Malicious code → halted and prevented from executing in the normal environment

- HIPSs can be tailored to the specific platform, or it can be a set of general-purpose tools may be used for a desktop or server system

- Some HIPS packages are designed to protect specific types of servers, such as Web servers and database servers
  - In this case, the HIPS looks for particular application attacks

Intrusion Prevention Systems
Network-Based IPS

- NIPS can detect and prevent intrusions on a network
  - It can modify or discard packets
  - Tear down a TCP connection

- It can be used for anomaly or misuse detection

- Applies filters to the full content of the flow every time a new packet arrives
  - Malicious flow → all subsequent packets belonging to the flow are dropped

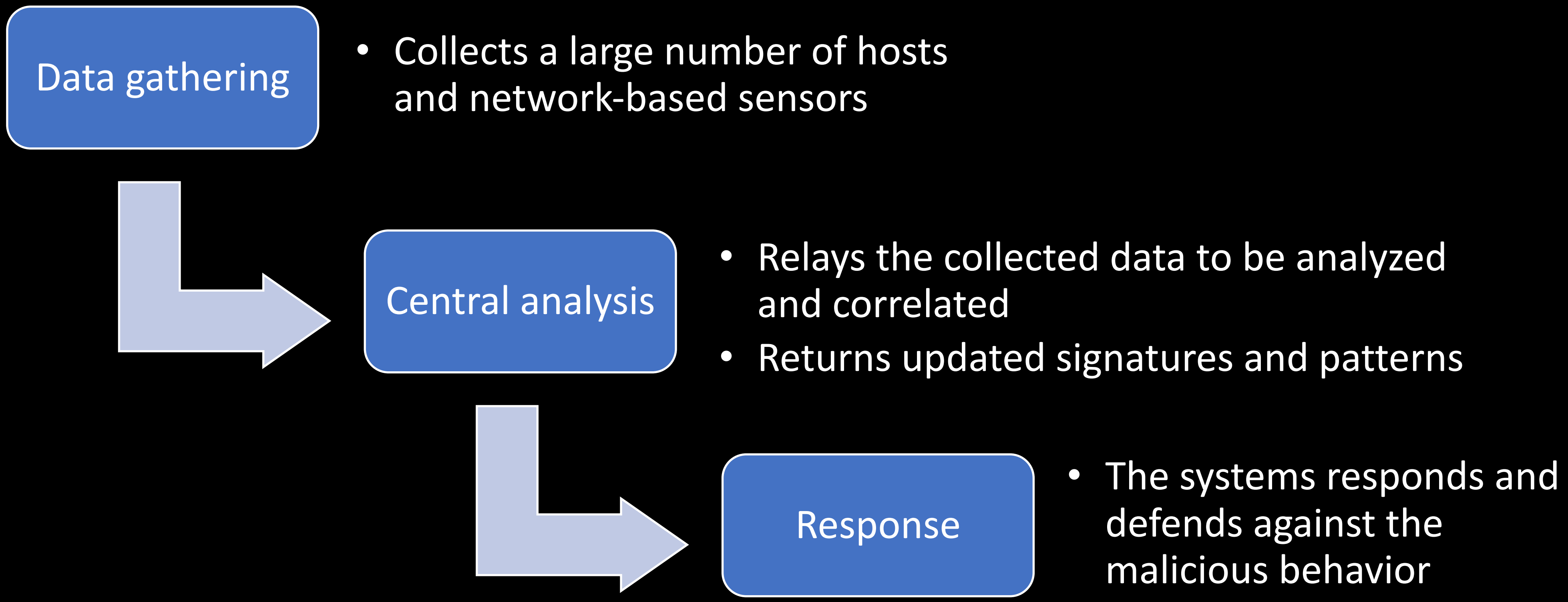
Intrusion Prevention Systems
Network-Based IPS

- NIPS applies the following methods to identify malicious packets:

Method	Description
Pattern matching	Scans incoming packets for specific byte sequences (the signature) stored in a database of known attacks
Stateful matching	Scans for attack signatures in the context of a traffic stream rather than individual packets
Protocol anomaly	Looks for deviation from standards set forth in RFCs
Traffic anomaly	Watches for unusual traffic activities, such as a **flood of UDP** packets or a new service appearing on the network
Statistical anomaly	Develops baselines of normal traffic activity and throughput, and alerts on deviations from those baselines

Intrusion Prevention Systems
Hybrid IPS

- Also known as **distributed IPS**
  - One of the such architecture is the digital immune systems – comprehensive system



Data gathering

- Collects a large number of hosts and network-based sensors

Central analysis

- Relays the collected data to be analyzed and correlated
- Returns updated signatures and patterns

Response

- The systems responds and defends against the malicious behavior

Content

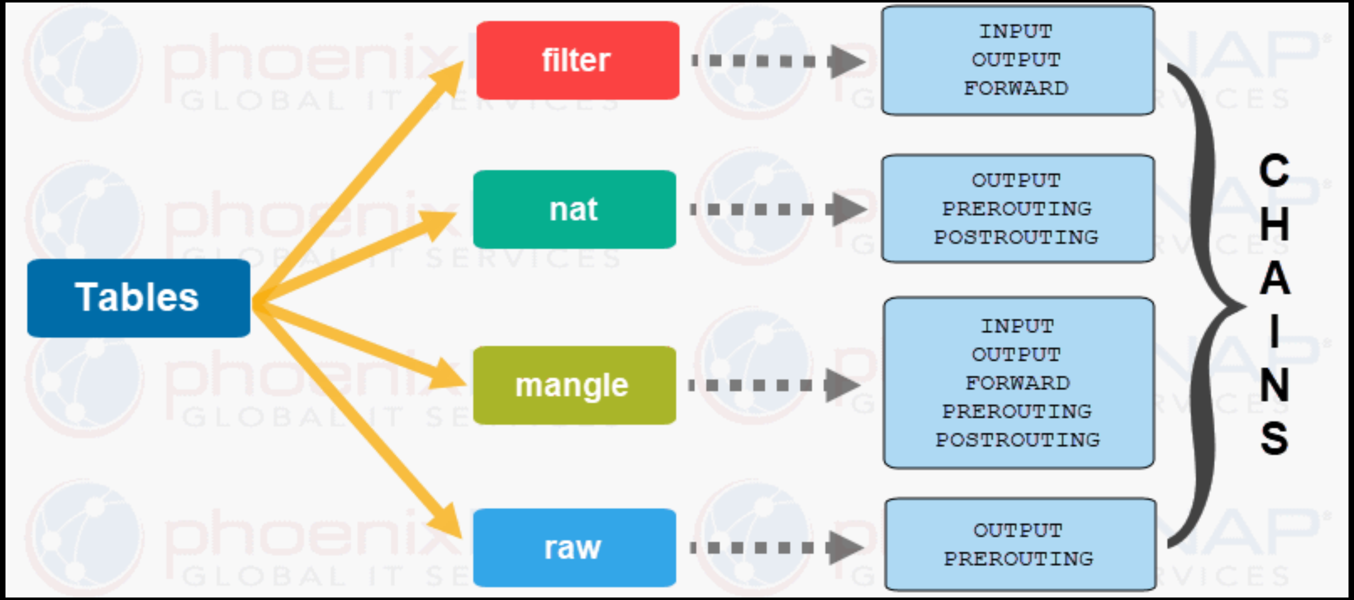
Firewalls, IDS, IPS
Firewalls
Intrusion Detection Systems
Intrusion Prevention Systems
Linux iptables
Linux iptables: Experimenting with Stateless Firewall Rules
Linux iptables: Connection Tracking and Stateful Firewall

Linux iptables

- Linux has a built-in firewall based on $netfilter$
  - Kernel's packet filtering framework
- This firewall is called $iptables$ – designed to filter + make changes to packets
- The kernel part implementation of the firewall is called $Xtables$
  - iptables is a user-space program to configure the firewall
- Other Linux firewalls:
  - UFW – Uncomplicated Firewall: developed to ease iptables firewall configuration
  - IPFire
  - OPNsense
  - pfSense

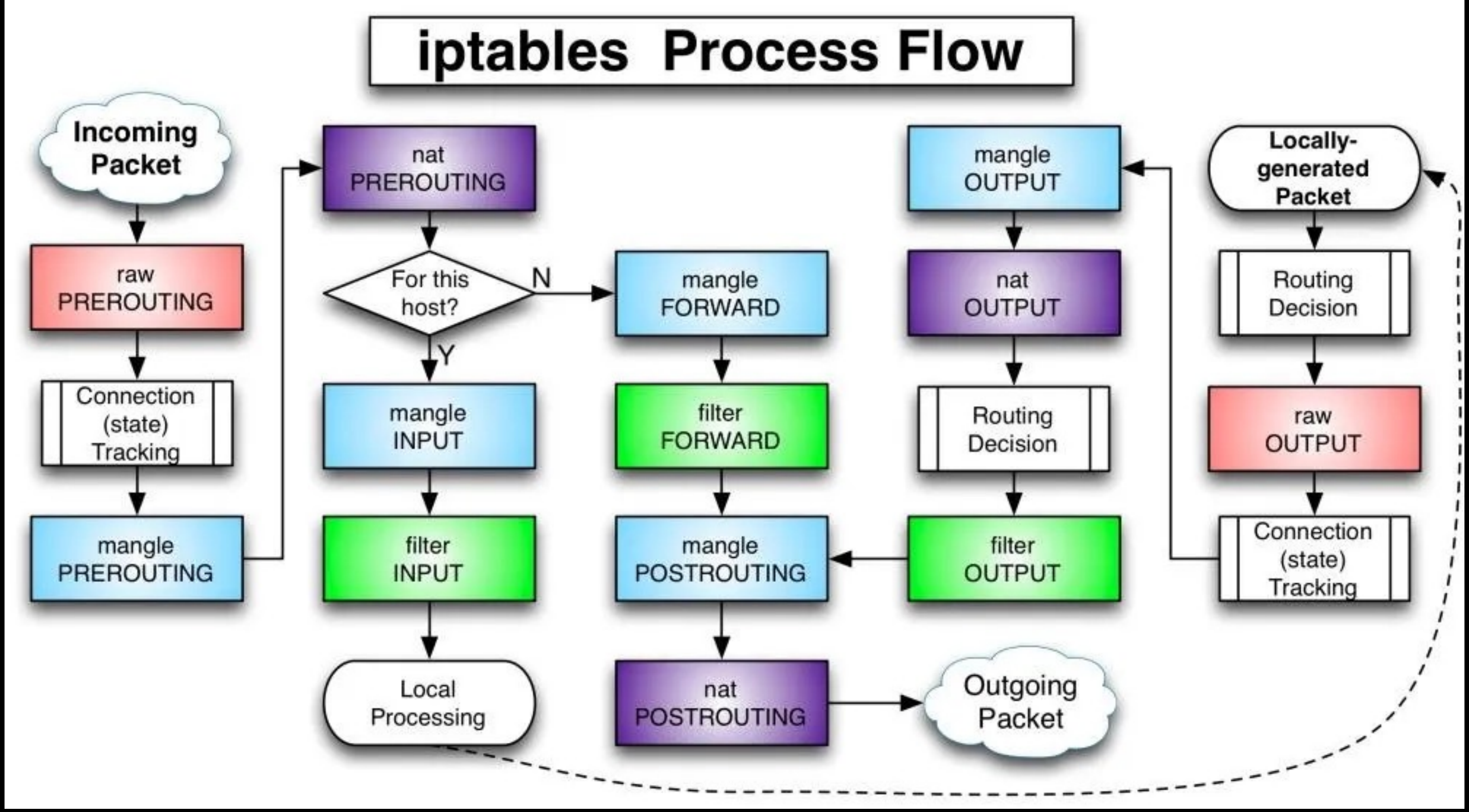
Linux iptables

- iptables is organized in a hierarchical structure: table, chain, and rules
  - $filter$: packet filtering
  - nat: modifying source or destination network addresses
  - $mangle$: packet content modification
  - raw: mark packets that should not be handled by the connection tracking system



Linux iptables

- Process flow in iptables



Linux iptables

- To add or remove rules, we need to specify:
    - The table (the default is $filter$)
    - Chain name
    - Operation on chain
- The general structure of the command

```
iptables -t   <table>   -<operation> <chain>   <rule>     -j <target>
            ----------- --------------------    -------    -----------
               Table            Chain           Rule         Action
```

Linux iptables

- List all the rules in a table (without line number)

```
iptables -t nat -L -n
```

- List all the rules in a table (with line number)

```
iptables -t filter -L -n --line-numbers
```

- Delete rule No. 2 in the INPUT chain of the filter table

```
iptables -t filter -D INPUT 2
```

- Replace a rule

```
iptables -t filter -R <chain> <rule-number> <new-rule-specification>
```

Linux iptables

- Drop all the incoming packets that satisfy the <rule>

```
iptables -t filter -A INPUT <rule> -j DROP
```

- To restore the filter table to its original state

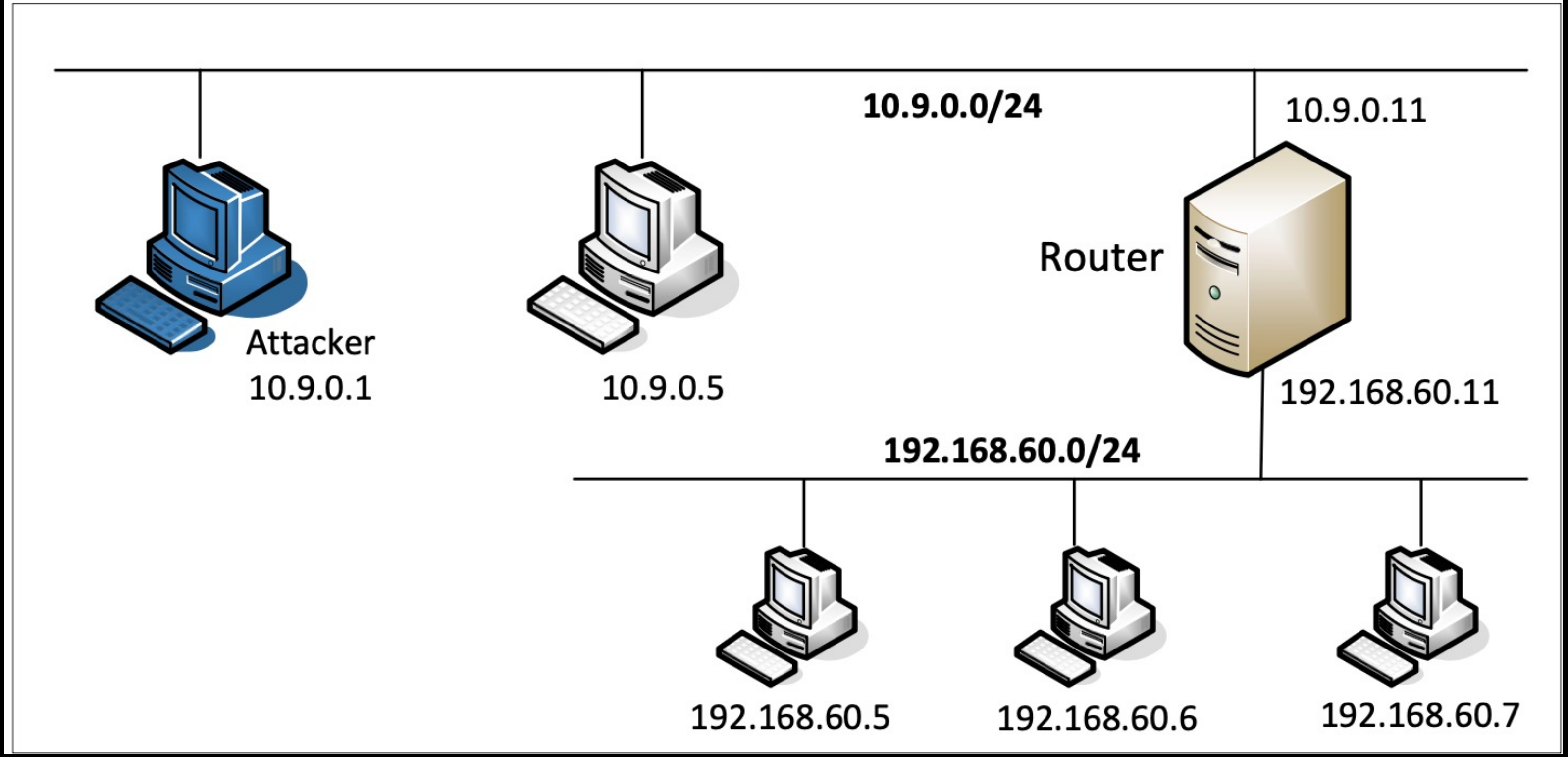
```
iptables -F
iptables -P OUTPUT ACCEPT
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
```

- Another way to restore the states of all the tables is to restart the container

```
docker restart <Container ID>
```

Linux iptables

- Lab setup



Content

Firewalls, IDS, IPS
Firewalls
Intrusion Detection Systems
Intrusion Prevention Systems
Linux iptables
Linux iptables: Experimenting with Stateless Firewall Rules
Linux iptables: Connection Tracking and Stateful Firewall

Linux iptables: Experimenting with Stateless Firewall Rules

Task 1: Protecting the Router

- Set up rules to prevent outside machines from accessing the router machine, except ping
  1. Execute the following commands on the router and then ping it from 10.9.0.5.

```
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
iptables -P OUTPUT DROP ← Set default rule for OUTPUT
iptables -P INPUT DROP  ← Set default rule for INPUT
```

  2. Can you telnet into the router?

Linux iptables: Experimenting with Stateless Firewall Rules

Task 2: Protecting the internal network

- Set up rules on the router to protect the internal network 192.168.60.0/24

1. Outside hosts cannot ping internal hosts
2. Outside hosts can ping the router
3. Internal hosts can ping outside hosts
4. Block everything else

- We need to use the FORWARD chain for this purpose
- Specify the direction, add the interface options using "-i xyz" (coming in from the xyz interface) and/or "-o xyz" (going out from the xyz interface)
  - Find out the interface names via the "ip addr" command

Linux iptables: Experimenting with Stateless Firewall Rules

Task 2: Protecting the internal network

1. Outside hosts cannot ping internal hosts
2. Outside hosts can ping the router
3. Internal hosts can ping outside hosts
4. Block everything else

```
iptables –P FORWARD DROP   // block everything else

//Outside hosts cannot ping internal hosts
iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-request -j DROP

//Internal hosts can ping outside hosts
iptables –A FORWARD –i eth1 -p icmp --icmp-type echo-request -j ACCEPT
iptables –A FORWARD –i eth0 -p icmp --icmp-type echo-reply -j ACCEPT
```

Linux iptables: Experimenting with Stateless Firewall Rules

Task 2: Protecting the internal network

- You can prevent hosts from pinging the router

```
iptables -A INPUT -p icmp --icmp-type echo-request –j DROP
```

Linux iptables: Experimenting with Stateless Firewall Rules

Task 2: Protecting internal servers

- Protect the TCP servers inside the internal network (192.168.60.0/24)

1. All the internal hosts run a telnet server (listening to port 23). Outside hosts can only access the telnet server on 192.168.60.5
2. Outside hosts cannot access other internal servers
3. Internal hosts can access all the internal servers
4. Internal hosts cannot access external servers

```
iptables –P FORWARD DROP   // block everything else

iptables -A FORWARD -i eth0 -p tcp -d 192.168.60.5 --dport 23 -j ACCEPT
iptables -A FORWARD -i eth1 -p tcp -s 192.168.60.5 --sport 23 -j ACCEPT
```

Content

	Firewalls, IDS, IPS
	Firewalls
	Intrusion Detection Systems
	Intrusion Prevention Systems
	Linux iptables
	Linux iptables: Experimenting with Stateless Firewall Rules
➡	Linux iptables: Connection Tracking and Stateful Firewall

Linux iptables: Connection Tracking and Stateful Firewall

- To support stateful firewalls, we need to be able to track connections
- This is achieved by the *conntrack* mechanism inside the kernel
- To list connection tracking

```
conntrack -L
```

- To display a real-time event log

```
conntrack -E
```

- To show the in-kernel connection tracking system statistics

```
conntrack -S
```

Linux iptables: Connection Tracking and Stateful Firewall

ICMP experiment:

- On the router:

```
conntrack -E
```

- On 10.9.0.5 ping 192.168.60.5

UDP experiment:

- On 192.168.60.5, start a netcat UDP server

```
nc -lu 9090
```

- On 10.9.0.5, send out UDP packets

```
nc -u 192.168.60.5 9090
<type something, then hit return>
```

Linux iptables: Connection Tracking and Stateful Firewall

TCP experiment:

- On 192.168.60.5, start a netcat TCP server

```
nc -l 9090
```

- On 10.9.0.5, send out TCP packets

```
nc 192.168.60.5 9090
<type something, then hit return>
```

Linux iptables: Connection Tracking and Stateful Firewall

- To make iptables work as a stateful firewall, use it with $conntrack$

- Pass the option “$-m\ conntrack$” to iptables to track the connections

- The “$- -ctstate\ ESTABLISHED, RELATED$” indicates that whether a packet belongs to an ESTABLISHED or RELATED connection

Linux iptables: Connection Tracking and Stateful Firewall

Task 3: allow external hosts to connect to internal servers using TCP

- On the router:

```
// drop everything
iptables -P FORWARD DROP

// allow TCP connections that are already established or belong to an existing connection
iptables -A FORWARD -p tcp -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

// The above rule doesn’t allow syn packet to initiate the connection, so allow it
iptables -A FORWARD -p tcp -i eth1 --dport 9090 --syn -m conntrack --ctstate NEW -j ACCEPT
```

- On 192.168.60.5:

```
nc –l 9090
```

On 10.9.0.5:

```
nc 192.168.60.5 9090
<type something, then hit return>
```

Linux iptables: Connection Tracking and Stateful Firewall

Task 4: allow internal hosts to connect to external servers using TCP

- On the router:

```
// drop everything
iptables -P FORWARD DROP

// allow TCP connections that are already established or belong to an existing connection
iptables -A FORWARD -p tcp -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

// Allow syn on both eth0 and eth1
iptables -A FORWARD -p tcp -i eth1 --dport 9090 --syn -m conntrack --ctstate NEW -j ACCEPT
iptables -A FORWARD -p tcp -i eth0 --dport 9090 --syn -m conntrack --ctstate NEW -j ACCEPT
```

- On 10.9.0.5 :

```
nc –l 9090
```

On 192.168.60.5 :

```
nc 10.9.0.5 9090
<type something, then hit return>
```

Linux iptables: Connection Tracking and Stateful Firewall

Task 5: Limiting network traffic passing through the firewall

- Limit how many packets from 10.9.0.5 are allowed to get into the internal network:

```
iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute --limit-burst 5 -j ACCEPT

iptables -A FORWARD -s 10.9.0.5 -j DROP
```