UNIVERSITY OF ŽILINA
Faculty of Management Science
and Informatics

Chapter 4:
Implementing Firewall Technologies

CCNA Security v2.0 / Network security v1.0
Chapter 4 / Modules 8 - 10

Bezpečnosť informačných sietí – KIS FRI UNIZA
Aktualizované v rámci projektu KEGA 026TUKE-4/2021.

Chapter Outline

- Access Control Lists
- Firewall Technologies
- IOS Zone-Based Policy Firewall
- Summary

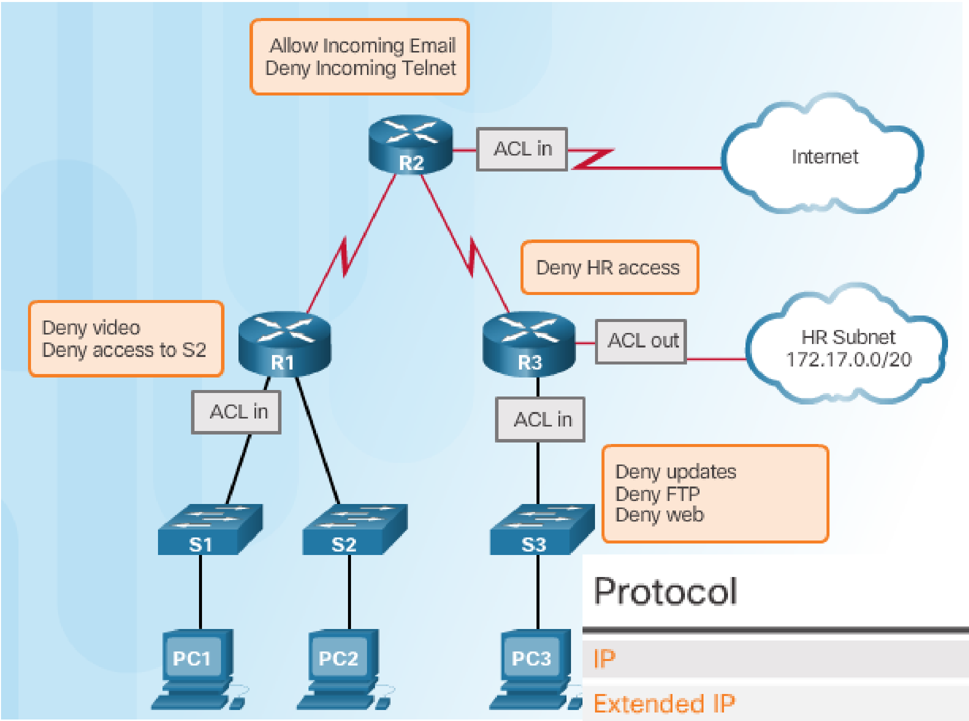
IPv4 Access Control List

Upon completion of this section, you should be able to:

- Configure standard and extended IPv4 ACLs using CLI.
- Use ACLs to mitigate common network attacks.
- Configure IPv6 ACLs using CLI.

Introduction to IPv4 Access Control Lists

- Widely used and familiar topic from CCNA R&S
  - sequential list of permit or deny statements
  - named as *access control entries* (ACEs)
- Works on L2 up to L7
- Historically identified by numbers
- Types:
  - **Standard/extended**
    - Ability to control source and destination of the traffic based on IP, protocol or port
  - **Named/numbered**



Protocol	Range
IP	1-99, 1300-1999
Extended IP	100-199, 2000-2699
Ethernet type code	200-299
DECnet and Extended DECnet	300-399
XNS	400-499
Extended XNS	500-599
AppleTalk	600-699
Ethernet address	700-799
IPX	800-899
Extended IPX	900-999
IPX SAP	1000-1099
Extended transparent bridging	1100-1199

Configuring Numbered and Named IPv4 ACLs

Standard Numbered ACL Syntax (control source only)

```
access-list {acl-#} {permit | deny | remark} source-addr [source-wildcard][log]
```

Extended Numbered ACL Syntax (control protocol, port, source and destination)

```
access-list acl-# {permit | deny | remark} protocol source-addr [source-wildcard]
dest-addr [dest-wildcard][operator port][established]
```

Named ACL Syntax

```
Router(config)# ip access-list [standard | extended] name_of_ACL
```

Standard ACE Syntax

```
Router(config-std-nacl)# {permit | deny | remark} {source [source-wildcard] | any}
```

Extended ACE Syntax

```
Router(config-ext-nacl)# {permit | deny | remark} protocol source-addr [source-wildcard]
dest-address [dest-wildcard] [operator port]
```

Revision CCNA – standard IPv4 ACL

```
access-list {acl-#} {permit | deny | remark} source-addr [source-wildcard][log]
```

Parameter	Description
acl-#	This is a decimal number from 1 to 99, or 1300 to 1999.
deny	Denies access if the conditions are matched.
permit	Permits access if the conditions are matched.
remark	Add a remark about entries in an IP access list to make the list easier to understand and scan.
source-addr	Number of the network or host from which the packet is being sent. There are two ways to specify the *source-addr* : • Use a 32-bit quantity in four-part, dotted-decimal format. • Use the keyword **any** as an abbreviation for a *source* and *source-wildcard* of 0.0.0.0 255.255.255.255.
source-wildcard	(Optional) 32-bit wildcard mask to be applied to the source. Places ones in the bit positions you want to ignore.
log	(Optional) Causes an informational logging message about the packet that matches the entry to be sent to the console. (The level of messages logged to the console is controlled by the **logging console** command.) The message includes the ACL number, whether the packet was permitted or denied, the source address, and the number of packets.
	The message is generated for the first packet that matches, and then at five-minute intervals, including the number of packets permitted or denied in the prior five-minute interval.

Revision CCNA – extended IPv4 ACL

```
access-list acl-# {permit | deny | remark} protocol source-addr [source-wildcard]
dest-addr [dest-wildcard][operator port][established]
```

Parameter	Description
acl-#	Identifies the access list using a number in the range 100 to 199 (for an extended IP ACL) and 2000 to 2699 (expanded IP ACLs).
deny	Denies access if the conditions are matched.
permit	Permits access if the conditions are matched.
remark	Used to enter a remark or comment.
protocol	Name or number of an Internet protocol. Common keywords include **icmp**, **ip**, **tcp**, or **udp**. To match any Internet protocol (including ICMP, TCP, and UDP) use the **ip** keyword.
source-addr	Number of the network or host from which the packet is being sent.
source-wildcard	Wildcard bits to be applied to source.
destination-addr	Number of the network or host to which the packet is being sent.
destination-wildcard	Wildcard bits to be applied to the destination.
operator	(Optional) Compares source or destination ports. Possible operands include **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), and **range** (inclusive range).
port	(Optional) The decimal number or name of a TCP or UDP port.
established	(Optional) For the TCP protocol only: Indicates an established connection.

Applying an IPv4 ACL

Syntax - Apply an ACL to an interface

```
Router(config)# interface TYPE SPEC
Router(config-if)# ip access-group {ACCESS-LIST-# | ACCESS-LIST-NAME} {in | out}
```

Syntax - Apply an ACL to the VTY lines

```
Router(config)# line vty  15
Router(config-if)# access-class {ACCESS-LIST-# | ACCESS-LIST-NAME} {in | out}
```

Example - Named Standard ACL

```
! Create ACL
R1(config)# ip access-list standard NO_ACCESS
R1(config-std-nacl)# deny host 192.168.10.10
R1(config-std-nacl)# permit any
R1(config-std-nacl)# exit
! Apply ACL
R1(config)#interface fa 0/1
R1(config-if)#ip access-group NO_ACCESS out
```

Example - Named Extended ACL

```
! Create a named ACL
R1(config)#ip access-list extended WEB-SERVICES-ONLY
R1(config-std-nacl)# remark Povol HTTP
R1(config-std-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config-std-nacl)# remark Povol HTTPS
R1(config-std-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq 443
! Apply ACL
R1(config)#int fa 0/0
R1(config-if)#ip access-group WEB-SERVICES-ONLY in
```

Applying an VTY IPv4 ACL (Cont.)

- Used to control a remote device access (telnet / SSH)

Syntax - Apply an ACL to the VTY lines

```
Router(config-line)# access-class {acl-#|name} {in|out}
```

Example - Named ACL on VTY lines with logging

```
R1(config)# ip access-list standard VTY_ACCESS
R1(config-std-nacl)# permit 192.168.10.10 log
R1(config-std-nacl)# deny any
R1(config-std-nacl)# exit
R1(config)# line vty 0 4
R1(config-line)# access-class VTY_ACCESS in
R1(config-line)# end
R1#
R1#!The administrator accesses the vty lines from 192.168.10.10
R1#
*Feb 26 18:58:30.579: %SEC-6-IPACCESSLOGNP: list VTY_ACCESS permitted 0
192.168.10.10 -> 0.0.0.0, 5 packets
R1# show access-lists
Standard IP access list VTY_ACCESS
    10 permit 192.168.10.10 log (6 matches)
    20 deny any
```

Editing Existing IPv4 ACLs

- By default all new ACEs are placed at the end of the list
- All ACL in modern IOSs are named now
  - Even numbered
  - Each ACEs has a number
  - Use line numbers to add/remove ACEs

Existing access list has three entries

```
Router# show access-lists
Extended IP access list 101
    10 permit tcp any any
    20 permit udp any any
    30 permit icmp any any
```

Access list has been edited, which adds a new ACE and replaces ACE line 20.

```
Router(config)# ip access-list extended 101
Router(config-ext-nacl)# no 20
Router(config-ext-nacl)# 5 deny tcp any any eq telnet
Router(config-ext-nacl)# 20 deny udp any any
```

Updated access list has four entries

```
Router# show access-lists
Extended IP access list 101
      5 deny tcp any any eq telnet
     10 permit tcp any any
     20 deny udp any any
     30 permit icmp any any
```

Sequence Numbers and Standard IPv4 ACLs

- Just a note:
  - host statements (those with specific IPv4 addresses and optionally without WildCard Mask) are listed first, but not necessarily in the order that they were entered
  - Internal Cisco logic is optimized for better search

Existing access list has four entries

```
router# show access-lists
Standard IP access list 19
    10 permit 192.168.100.1
    20 permit 10.10.10.0, wildcard bits 0.0.0.255
    30 permit 201.101.110.0, wildcard bits 0.0.0.255
    40 deny any
```

Access list has been edited, which adds a new ACE that permits a specific IP address.

```
router(config)# ip access-list standard 19
router(config-std-nacl)# 25 permit 172.22.1.1
```

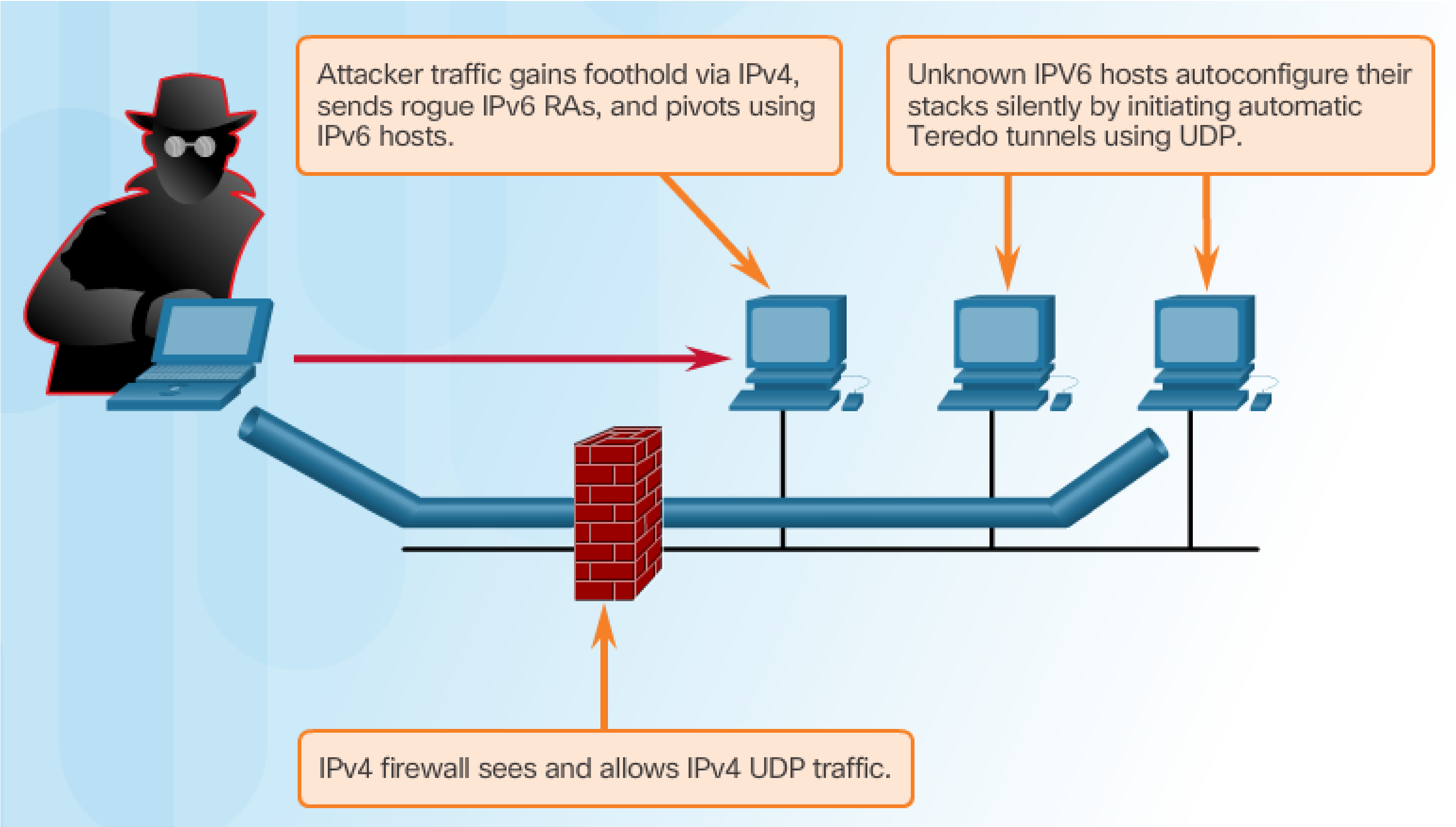
Updated access list places the new ACE before line 20

```
router# show access-lists
Standard IP access list 19
    10 permit 192.168.100.1
    25 permit 172.22.1.1
    20 permit 10.10.10.0, wildcard bits 0.0.0.255
    30 permit 201.101.110.0, wildcard bits 0.0.0.255
    40 deny any
```

IPv6 ACLs

Introducing IPv6 ACLs

- IPv6 and dual stack world
  - IPv6 attacks are becoming more pervasive
  - Some networks protect IPv4 but forgot on IPv6
    - once the IPv6 was activated
  - IPv4 could be a security hole to attack on IPv6
    - Penetrate through IPv4 and then attack IPv6 segments
    - Tunneling IPv6 within IP4
      - Pass-through IPv4 firewalls



ACE Syntax

IPv6 ACL

```
R1(config)# ipv6 access-list access-list-name
R1(config-ipv6-acl)# deny | permit protocol {source-ipv6-prefix/prefix-length | any | host
source-ipv6-address} [operator [port-number]] {destination-ipv6-prefix/prefix-length | any |
host destination-ipv6-address} [operator [port-number]]
```

Parameter	Description
`deny	permit`	Specifies whether to deny or permit the packet.
`protocol`	Enter the name or number of an Internet protocol, or an integer representing an IPv6 protocol number.
`source-ipv6-prefix/prefix-length` `destination-ipv6-address`	The source or destination IPv6 network or class of networks for which to set deny or permit conditions.
`any`	Enter `any` as an abbreviation for the IPv6 prefix ::/0. This matches all addresses.
`host`	For `host source-ipv6-address` or `destination-ipv6-address`, enter the source or destination IPv6 host address for which to set deny or permit conditions.
`operator`	(Optional) An operand that compares the source or destination ports of the specified protocol. Operands are lt (less than), gt (greater than), eq (equal), neq (not equal), and range.
`port-number`	(Optional) A decimal number or the name of a TCP or UDP port for filtering TCP or UDP, respectively.

Syntax - Apply an IPV6 ACL to an interface

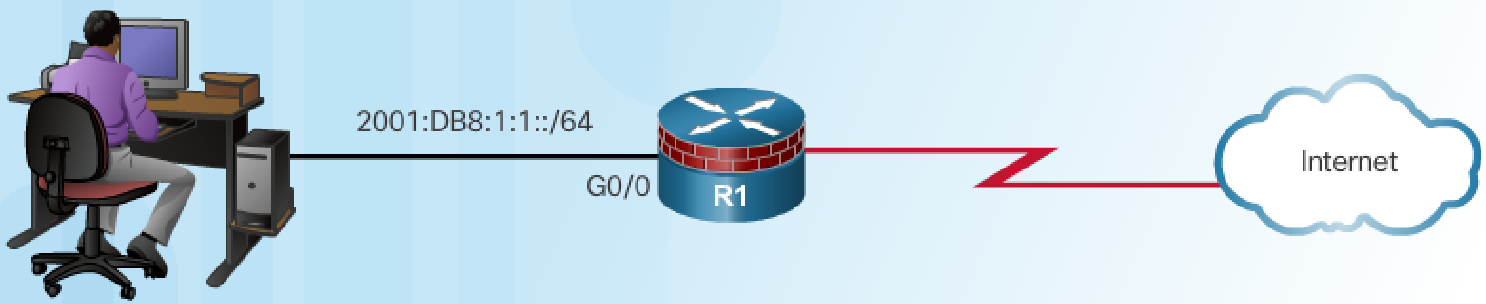
```
Router(config)# interface TYPE SPEC
Router(config-if)# ipv6 traffic-filter {ACCESS-LIST-# | ACCESS-LIST-NAME} {in | out}
```

Syntax - Apply an IPv6 ACL to the VTY lines

```
Router(config)# line vty  15
Router(config-if)# ipv6 access-class {ACCESS-LIST-# | ACCESS-LIST-NAME} {in | out}
```

Configuring IPv6 ACLs

- Differences against IPv4 ACL
  - IPv6 has only named extended ACL
  - Uses prefix-lists not wildcards
  - Has some additional implicit permit conditions (NDP)
    - ICMP neighbor advertisement/solicitation (DAD + IPv6 to MAC mapping)
      - permit icmp any any **nd-na**
      - permit icmp any any **nd-ns**

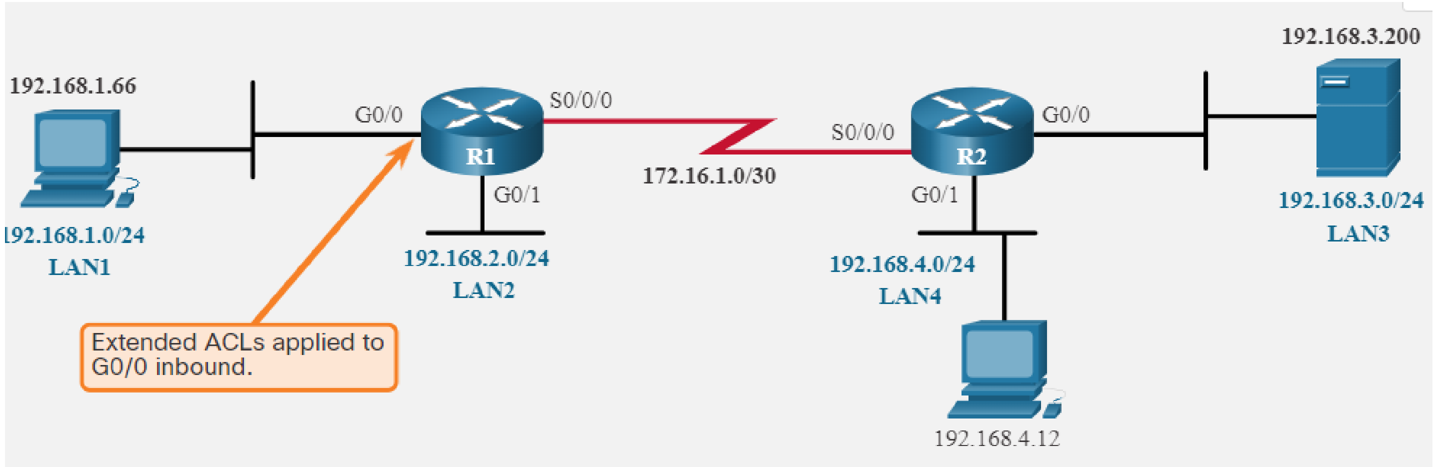



```
R1(config)# ipv6 access-list LAN_ONLY
R1(config-ipv6-acl)# permit 2001:db8:1:1::/64 any
R1(config-ipv6-acl)# permit icmp any any nd-na
R1(config-ipv6-acl)# permit icmp any any nd-ns
R1(config-ipv6-acl)# deny ipv6 any any
R1(config-ipv6-acl)# end
R1# show ipv6 access-list
IPv6 access list LAN_ONLY
    permit ipv6 2001:DB8:1:1::/64 any sequence 10
    permit icmp any any nd-na sequence 20
    permit icmp any any nd-ns sequence 30
    deny ipv6 any any sequence 40
R1#
```

ACL Configuration Guidelines

- Create an ACL globally and then apply it.
- Ensure the last statement is an implicit `deny any` or `deny any any`.
- Remember that statement order is important because ACLs are processed top-down. As soon as a statement is matched the ACL is exited.
- Ensure that the most specific statements are at the top of the list.
- Remember that only one ACL is allowed per interface, per protocol, per direction.
- Remember that new statements for an existing ACL are added to the bottom of the ACL by default.
- Remember that router generated packets are not filtered by outbound ACLs.
- Place standard ACLs as close to the destination as possible.
- Place extended ACLs as close to the source as possible.

Q/A




```
access-list 105 permit tcp 192.168.1.0 0.0.0.255 host 192.168.3.200 eq 80
access-list 105 permit ip host 192.168.1.66 host 192.168.3.200
access-list 105 permit tcp 192.168.1.0 0.0.0.255 host 192.168.4.12 eq 22
access-list 105 permit tcp host 192.168.1.66 192.168.2.0 0.0.0.255 eq 23
```

Source	Destination	Protocol
192.168.1.67	192.168.2.88	http
192.168.1.66	192.168.4.12	ssh
192.168.1.77	192.168.3.75	http
192.168.1.66	192.168.2.75	telnet
192.168.1.77	192.168.2.75	telnet
192.168.1.66	192.168.3.200	telnet

Permit or deny?

Mitigating Attacks with ACLs

Case 1.

Antispoofing with ACLs

- Spoofing
  - Uses source IP addresses of attack targets
  - DoS/DDoS use IP spoofing very often
- Simple idea
  - Permit only IP addresses used in my network
  - Or
  - Deny unused well known IP address ranges
    - Private, mcast, broadcast, localhost

Inbound on S0/0/0

```
R1(config)# access-list 150 deny ip 0.0.0.0 255.255.255.255 any
R1(config)# access-list 150 deny ip 10.0.0.0 0.255.255.255 any
R1(config)# access-list 150 deny ip 127.0.0.0 0.255.255.255 any
R1(config)# access-list 150 deny ip 172.16.0.0 0.15.255.255 any
R1(config)# access-list 150 deny ip 192.168.0.0 0.0.255.255 any
R1(config)# access-list 150 deny ip 224.0.0.0 15.255.255.255 any
R1(config)# access-list 150 deny ip host 255.255.255.255 any
```

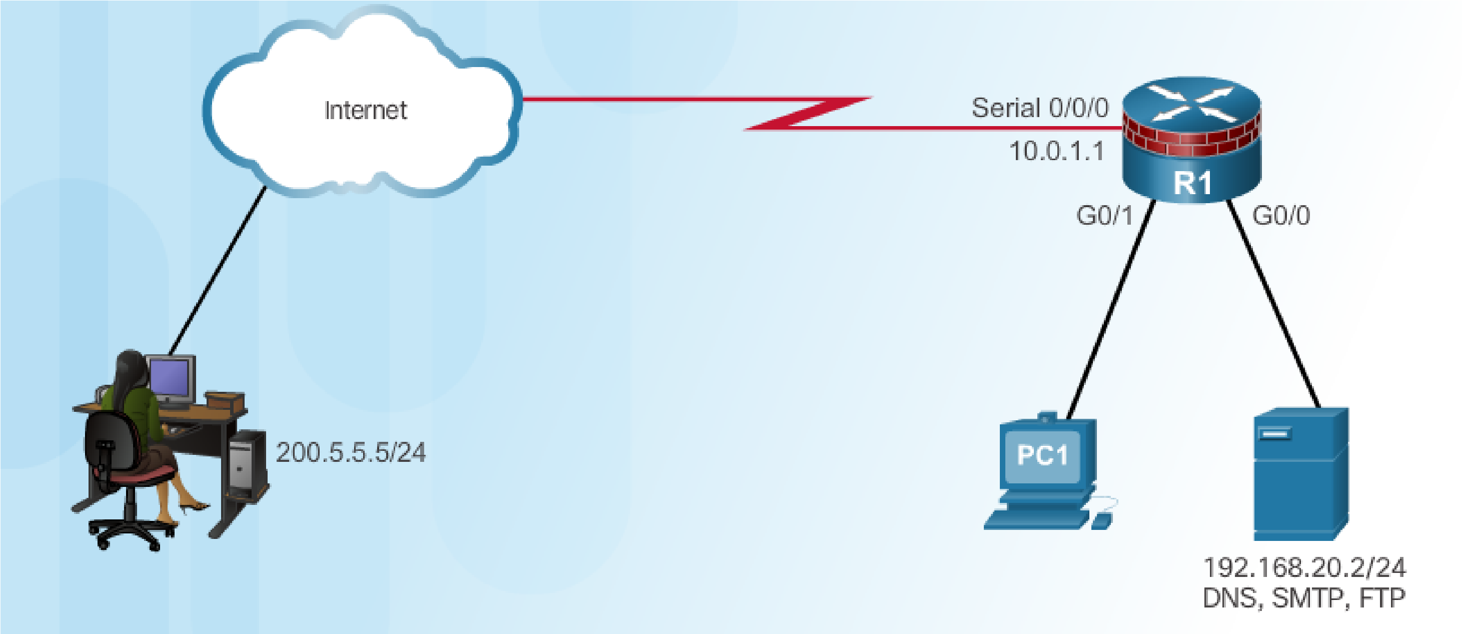
Inbound on G0/0

```
R1(config)# access-list 105 permit ip 192.168.1.0 0.0.0.255 any
```

Case 2.

Permitting Necessary Traffic through a Firewall

- Kind of policy
- => Explicitly permit only **certain** and **required** types of traffic
  - DNS, SMTP, SSH, HTTP/S, SNMP, syslog
- Force to use **specific servers**
  - Company DNS, NTP and no any other



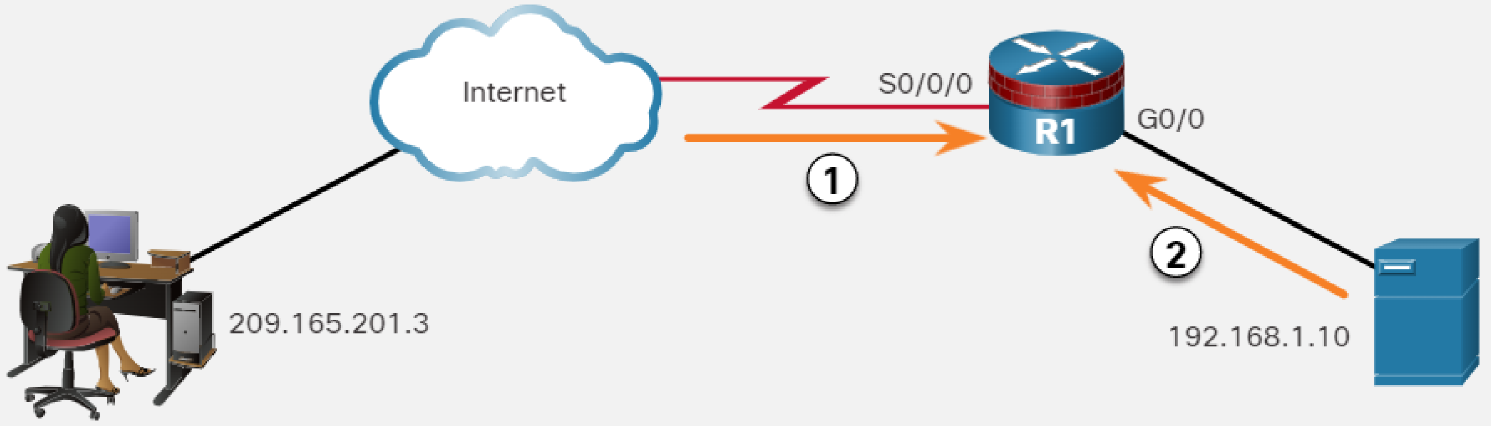
Inbound on Serial 0/0/0

```
R1(config)# access-list 180 permit udp any host 192.168.20.2 eq domain
R1(config)# access-list 180 permit tcp any host 192.168.20.2 eq smtp
R1(config)# access-list 180 permit tcp any host 192.168.20.2 eq ftp
R1(config)# access-list 180 permit tcp host 200.5.5.5 host 10.0.1.1 eq 22
R1(config)# access-list 180 permit udp host 200.5.5.5 host 10.0.1.1 eq syslog
R1(config)# access-list 180 permit udp host 200.5.5.5 host 10.0.1.1 eq snmptrap
```

Case 3.

Mitigating ICMP Abuse

- ICMP
  - Required for network operation
    - Echo/echo reply, source quench, unreachable, parameter problem, packet too big
  - But used for attacks too
    - discovery
    - DoS flood
    - route redirects
- Mitigation
  - Block all unnecessary ICMP message types as is required



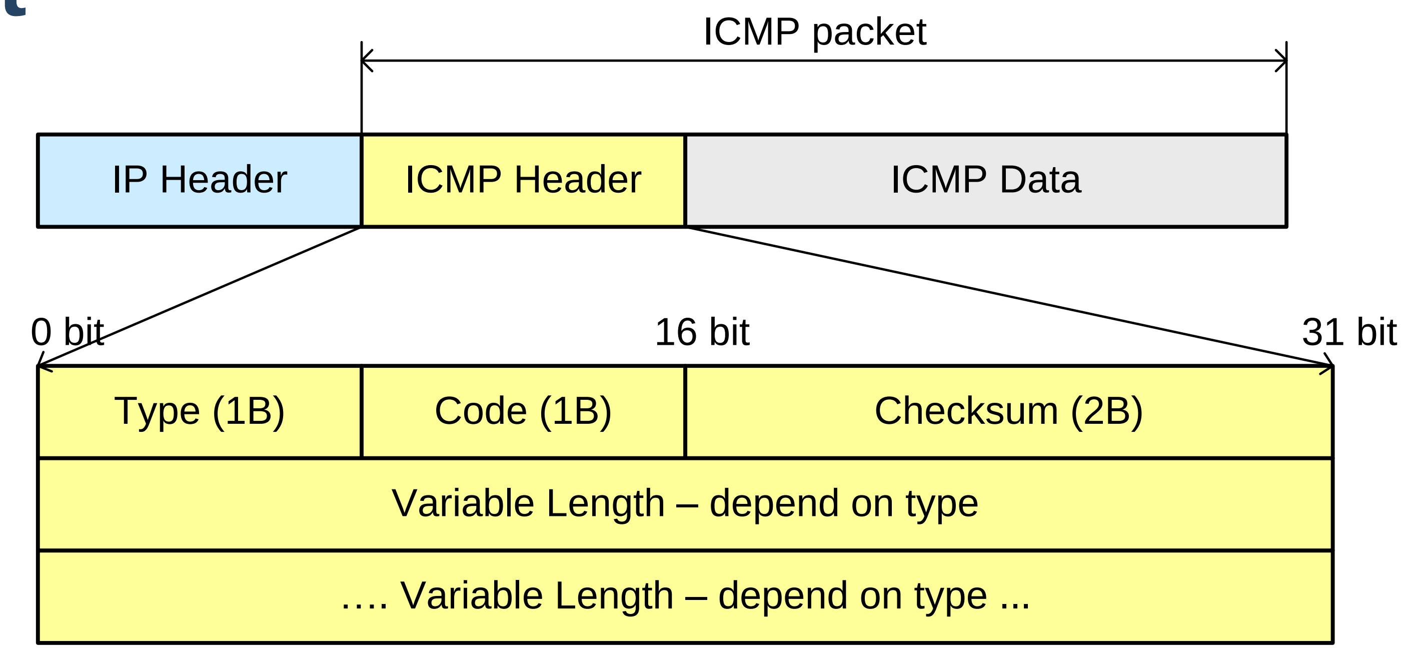
1. Rules on R1 for ICMP traffic from the Internet

```
access-list 112 permit icmp any any echo-reply
access-list 112 permit icmp any any source-quench
access-list 112 permit icmp any any unreachable
access-list 112 deny icmp any any
access-list 112 permit ip any any
```

2. Rules on R1 for ICMP traffic from inside the network

```
access-list 114 permit icmp 192.168.1.0 0.0.0.255 any echo
access-list 114 permit icmp 192.168.1.0 0.0.0.255 any parameter-problem
access-list 114 permit icmp 192.168.1.0 0.0.0.255 any packet-too-big
access-list 114 permit icmp 192.168.1.0 0.0.0.255 any source-quench
access-list 114 deny icmp any any
access-list 114 permit ip any any
```

ICMP paket



- Type
  - Udáva typ ICMP správy
- Code
  - Každý typ môže mať viac druhov, ktoré bližšie špecifikujú čo ICMP reportuje
- Checksum
  - Kontrolná suma
- Type + Code + Checksum
  - Majú všetky ICMP správy
- Variable Length
  - Mení sa podľa Type a Code

Typy ICMP správ

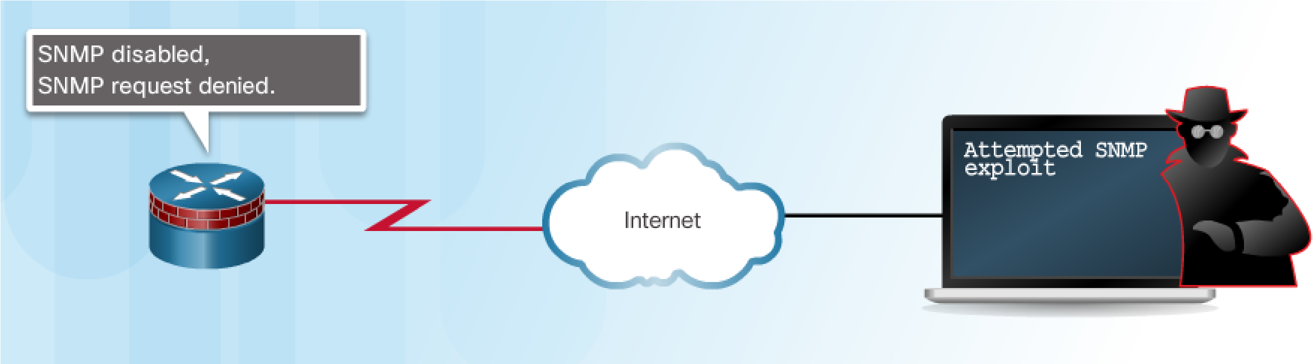
- Typ
  - 0 Echo Reply
  - 3 Destination Unreachable
    - Report problému s doručením
    - 0 = net unreachable;
    - 1 = host unreachable;
    - 2 = protocol unreachable;
    - 3 = port unreachable;
    - 4 = fragmentation needed and DF set;
    - 5 = source route failed.
  - 4 Source Quench
  - 5 Redirect / Change request
  - 8 Echo Request
  - 9 Router Advertisment

- 10 Router Solicitation
- 11 Time Exceed
- 12 Parameter problem
- 13 Timestamp Request
- 14 Timestamp Reply
- 15 Information Request
- 16 Information Reply
- 17 Address Mask Request
- 18 Address Mask Reply

Error testing and error reporting: ICMP 0, 3, 8, 11

Control ICMP messages: zvyšok

Mitigating SNMP Exploits



- SNMP is required
  - Set and apply an ACL blocking SNMP queries on outbound interfaces
  - Or configure SNMP allowing SNMP access only from snmp managers
- SNMP is not required
  - Turn off the service
    - **no snmp-server**

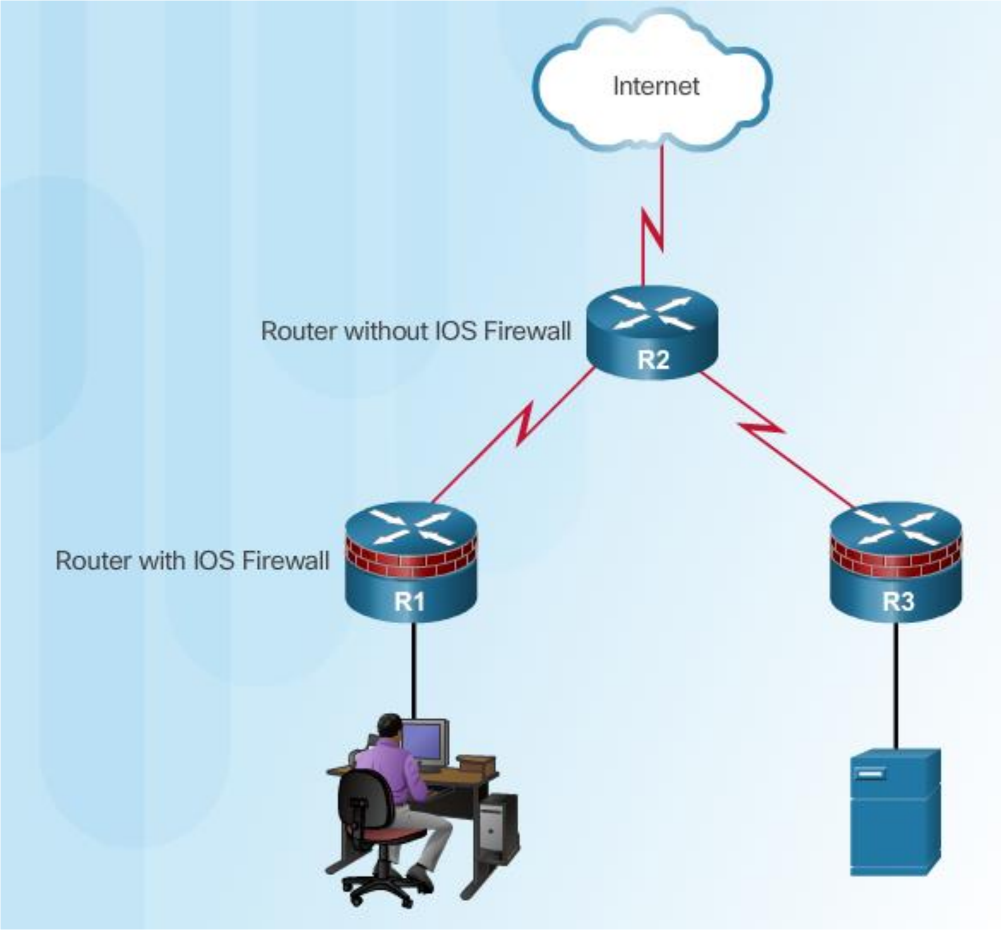
Firewall Technologies

Upon completion of this section, you should be able to:

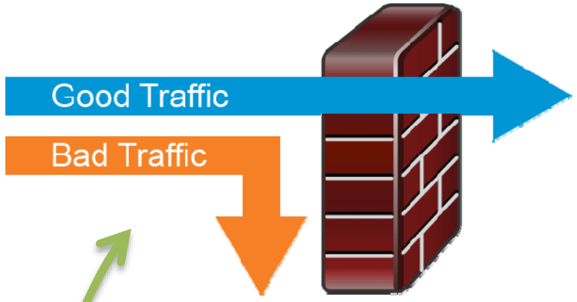
- Explain how firewalls are used to help secure networks.
- Describe the various types of firewalls.
- Configure a classic firewall.
- Explain design considerations for implementing firewall technologies.

Defining Firewalls

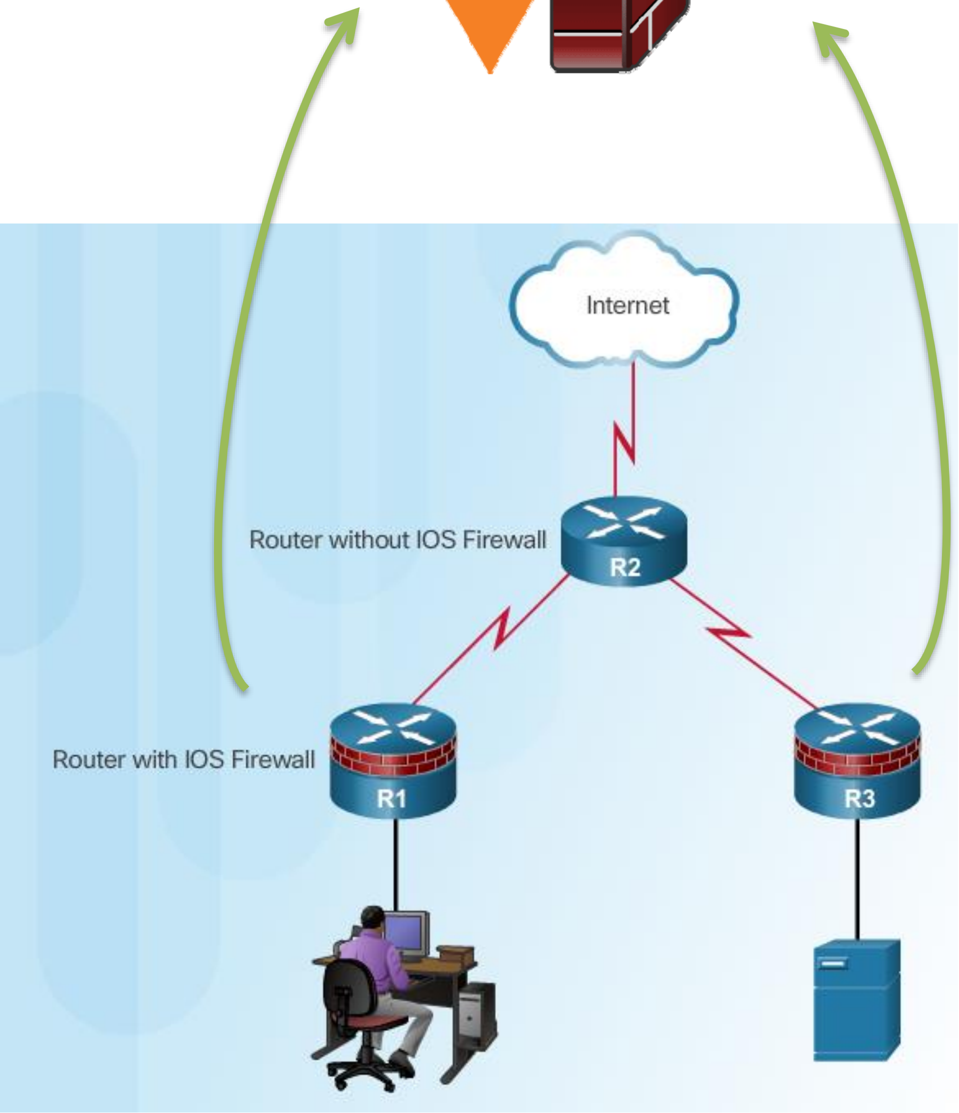
- Firewall = Fireproof wall (fire protection terminology)
  - An obstacle deployed between two networks
  - 1988 – DEC - first packet stateless FW
  - 1989 - AT&T - first stateful FW
  - Originally
    - Router or server with firewall sw functionalities
  - Now
    - **A system or group of systems that enforces an access control policy between networks**
    - Realized as:
      - Standalone hw or sw appliance
      - Router or switch with statefull fw. functionalities



Defining Firewalls – expected features



- All firewalls' requirements (possibly):
  - **FW must be resistant to attack**
    - Secured OS, secure policies
    - If the firewall is compromised = all the security functions it provides are also compromised.
  - **FW must be the only transit point between IN/OUT networks and all traffic flows through the firewall**
    - By the design: all other connection to network except through FW are blocked
  - **FW allows (permit) only defined traffic, undefined is blocked (drop/deny)**
    - Enforce the access control policy (ACL rules)



Common Benefits and Limitations of Firewalls

Benefits

- Sanitize protocol flow, which prevents the exploitation of protocol flaws.
- Prevent the exposure of sensitive hosts, resources, and applications to untrusted users.
- Block malicious data from servers and clients.
- Reduce security management complexity by off-loading most of the network access control to a few firewalls in the network
- Platform for other network perimeter functions
  - NAT, IPsec GW,

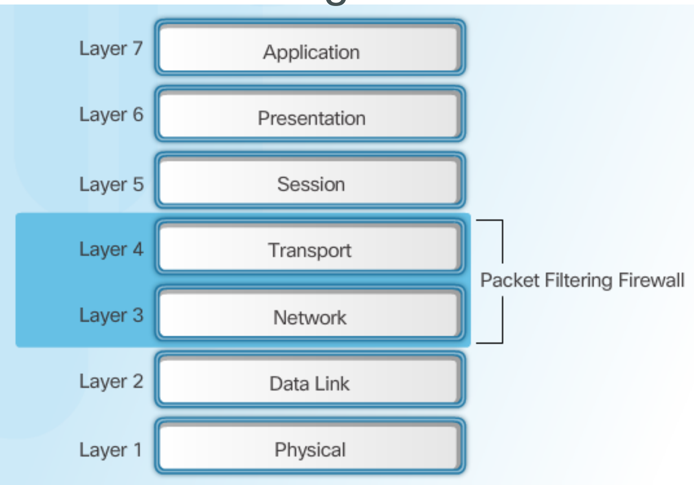
Limitations

- Firewall concentrates security functions
  - Misconfiguration has disastrous consequences
- Single point of failure
  - In case of failures and misconfiguration firewall can have serious consequences for the network
- Network bottleneck
  - Network performance can slow down
- Cannot provide full protection, for example for inside threats
  - Users might proactively search for ways around the firewall to receive blocked material, which exposes the network to potential attack.
  - Unauthorized traffic can be tunneled or hidden as legitimate traffic through the firewall.

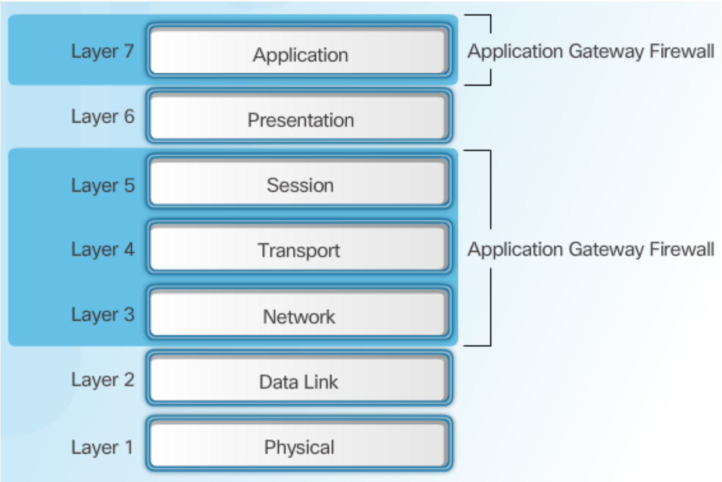
Network Firewall Types

- **Packet filtering firewall**
  - Typically router with limited filtering at L3/L4
- **Stateful firewall**
  - Has a knowledge on the state of connections
    - Who initiate and terminate session,
    - which data packet belongs to which session
- **Application gateway firewall (proxy firewall)**
  - Works up to L7
  - Usually realized in sw.
- **Other classification**
  - Transparent firewall
  - Hybrid firewall
  - “Next Gen firewalls”
  - NAT firewall
  - ---
  - Host-based (server and personal) firewall

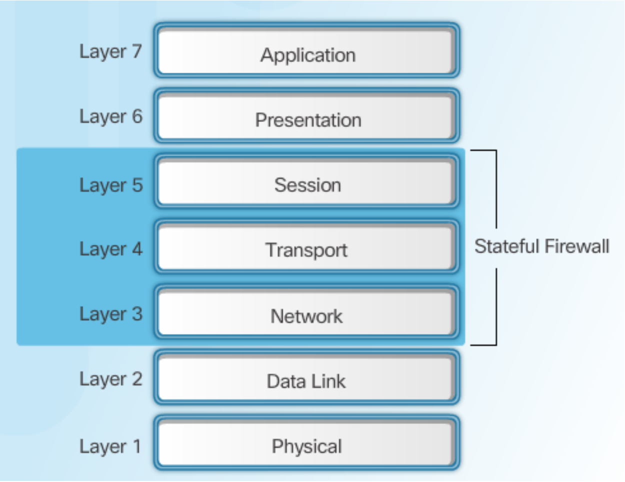
Packet Filtering Firewall



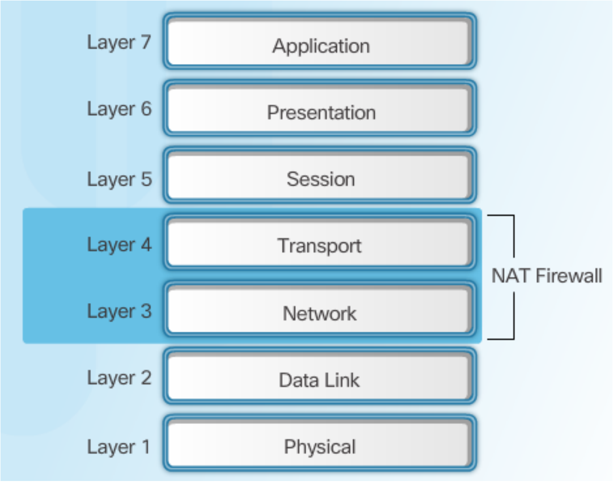
Application Gateway Firewall



Stateful Firewall

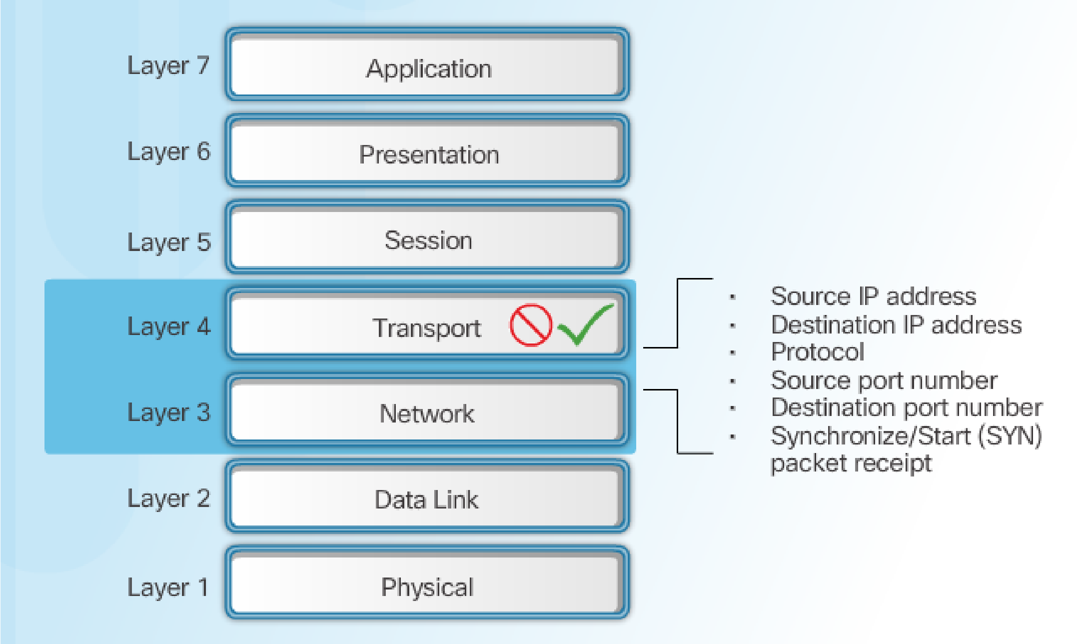


NAT Firewall



Packet Filtering Firewall - Benefits & Limitations

- Packet Filtering Firewall
  - Usually, part of a router firewall
  - Stateless
    - Think on two directions
    - Unable to distinguish directions and packet relationship
  - Analyze packets on its static header content
    - **5tuples**: addresses, ports, protocol
  - Simple policy with table lookup



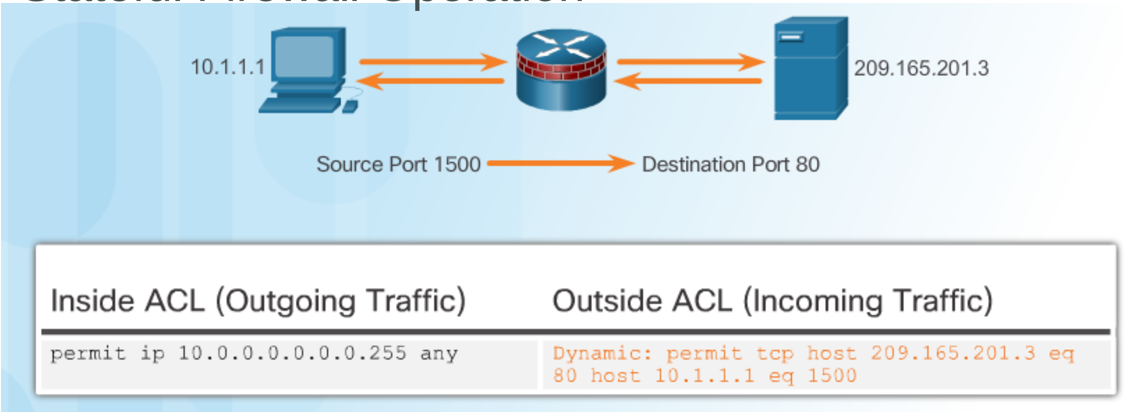
- **Benefits**
  - Simple permit/deny actions (for example ACL on Cisco devices)
  - Low impact on network performance
  - Easy to implement, widely supported
  - Usefull for initial degree of protection
  - Perform almost all the tasks of a high-end firewall at a much lower cost

- **Limitations**
  - Malicious packets that meet ACL criteria will pass
  - Do not reliably filter fragmented packets
    - i.e. deny packet with TCP header inside, but other fragments pass
  - Uses complex ACLs, which can be difficult to implement and maintain
  - Are not able to dynamically filter certain services
    - Sessions with dynamic port negotiation
  - Are stateless
    - examines each packet individually as are unable to determine if a packet is part of connection

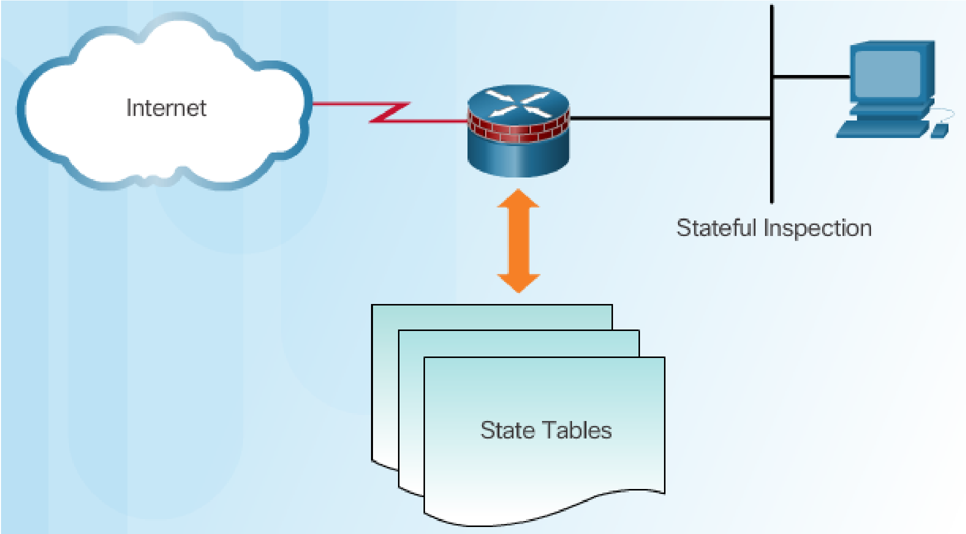
Statefull Firewalls

- An evolution of packet filter => Works on L2 – L4 (L5)
- Statefull means
  - Analyze TCP headers and protocol Three Way Handshake (Beginning: SYN, SYN/ACK, ACK, Ending: Fin/ACK)
  - Tracks and keeps individual connection information on all interfaces and directions
    - For example, analyze the initialization of the flow
    - Build state flow table
  - And utilize this info on filtering decisions
    - For example, for allowing returned traffic
- The most versatile and the most common firewall technologies in use
  - Usually, basic FW feature
    - FW only with limited features available
    - FW without licenses

Stateful Firewall Operation



State Tables



Stateful Firewall - Benefits and Limitations

Benefits

- Primary means of defense by filtering unwanted, unnecessary, or undesirable traffic
- Strong packet filtering with more stringent control
- Improved performance over packet filters
- Defends against spoofing and DoS attack
  - Able to determine if packet belongs to a valid session
- Provides more log information
- Simpler configuration
  - ACL is used to defined outgoing traffic, do not need to think on opposite direction

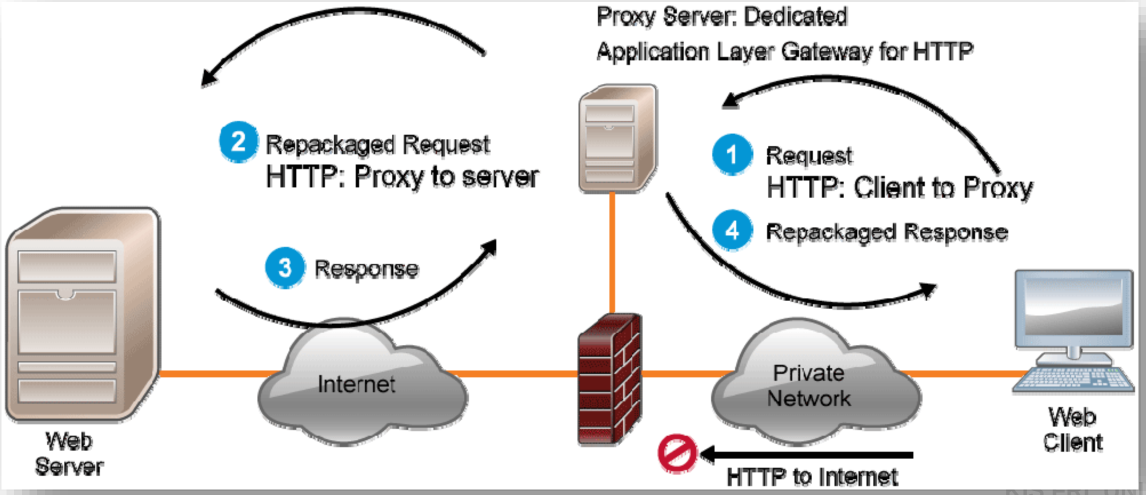
Limitations

- No Application Layer inspection and attack protection
- Limited tracking of stateless protocols
  - Works better for stateful TCP sessions
  - Problem with UDP flows
- May be difficult to detect dynamic port negotiation (some FW may support)
  - VoIP (SIP/SDP), FTP (control and data channel)
- Does not support user authentication

Application Gateway Firewall (Proxy)

- Further evolution
  - Considered more secure as stateful FW
- Also know as
  - Application **proxy**
  - Application layer gateway (ALG)
- Works at application layer (OSI L7)
  - Support L7 protocol analysis
  - May work with user IDs
  - May work as service broker
    - Requests user authentication data
    - Using his credentials to contact an app server
  - Does not act as L3 forwarder, it terminates L3
  - and separates application sessions (for example https)

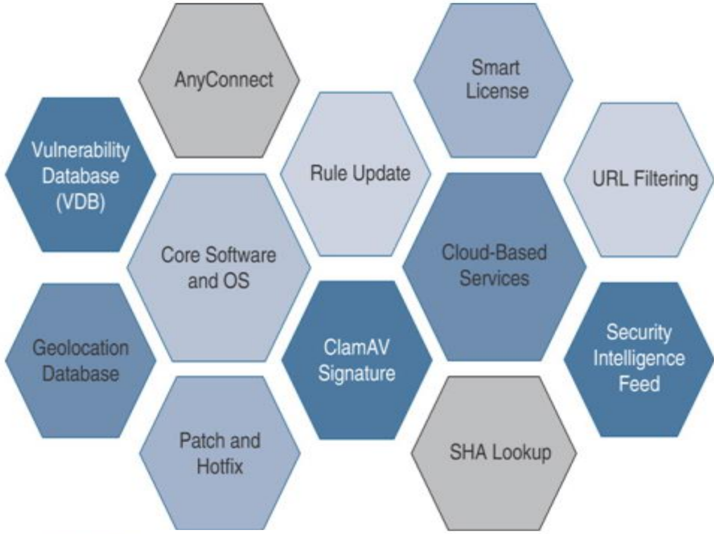
- Limitations
  - Main limitation is increased computational complexity that require more resources
  - Historically specially purposed appliance per service/application
    - Now are some features part of Next Gen Firewalls



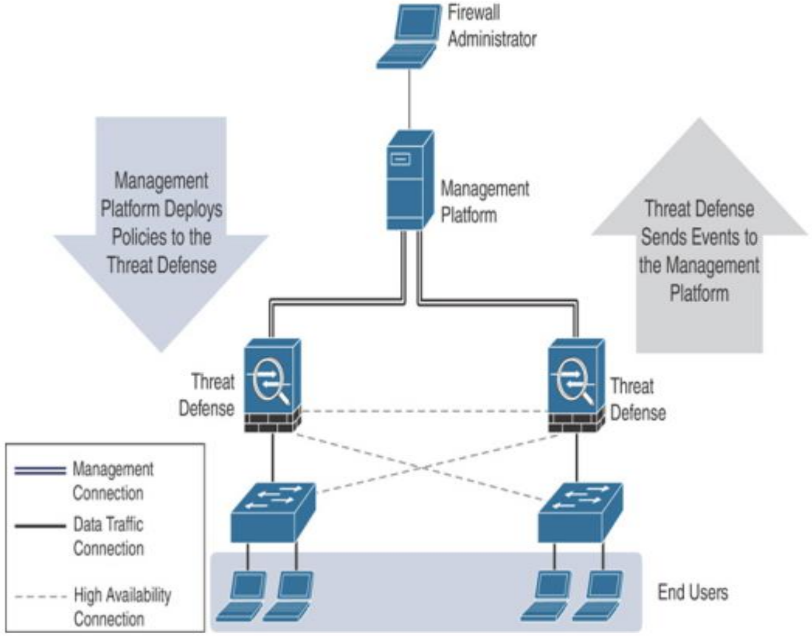
Next-gen firewall

- Brings stateful and application firewall improvements
  - Application Visibility and Control
    - Granular identification, visibility, and control of behaviors within applications
    - App filtering based on IP address and domains reputation
  - Context awareness
    - Who is going where, when and from
    - User ID functionalities, enforcement of policies based on the user, device, role, application type, and threat profile
  - Integrated intrusion prevention systems (IPS), SSL inspection, blacklisting, URL filtering, Antimalware protection, ….
  - Performance of NAT, VPN …

- **Cisco Next Gen FW = Cisco Secure Firewall (correct name now)**
  - Previous name **Cisco Firepower Threat Defense** (FTD) = Sourcefire's FirePOWER services + Snort + Cisco Adaptive Security Appliance (ASA)



Cisco Firepower Threat Defense (FTD)



- **Cisco Firepower Threat Defense => different management possibilities**
  - Secure Firewall Device Manager (FDM)
    - Allows you to manage a single threat defense locally
    - It is built in device manager
  - Secure Firewall Management Center (FMC)
    - Allows you to manage multiple threat defenses (hundreds) from a centralized location
  - Cisco Defense Orchestrator (CDO)
    - Cloud-delivered management platform (Cisco SaaS)

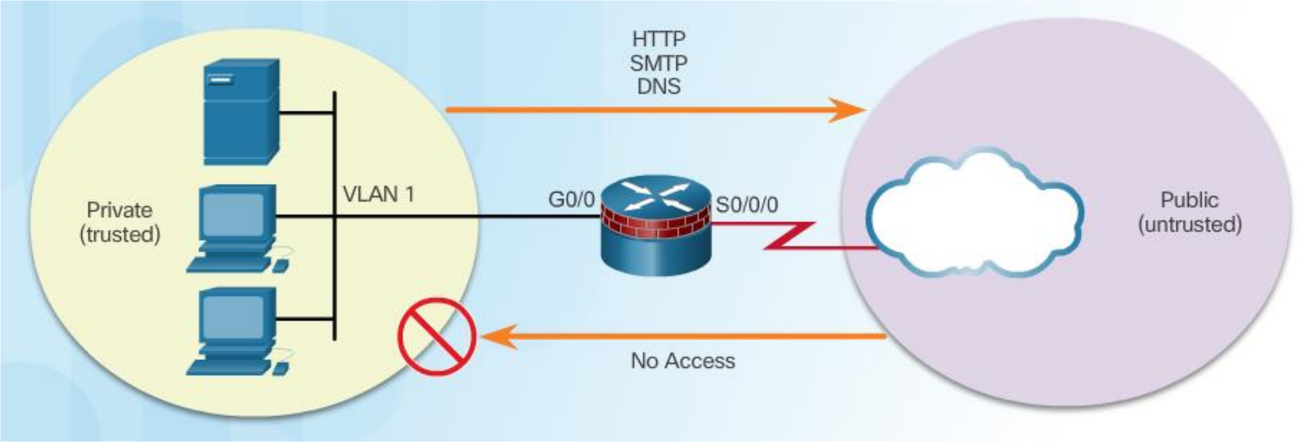
NextGen Firewall Magic Quadrant

2021

Firewalls in Network Design

Note: Courses focuses on the use of routers as firewalls

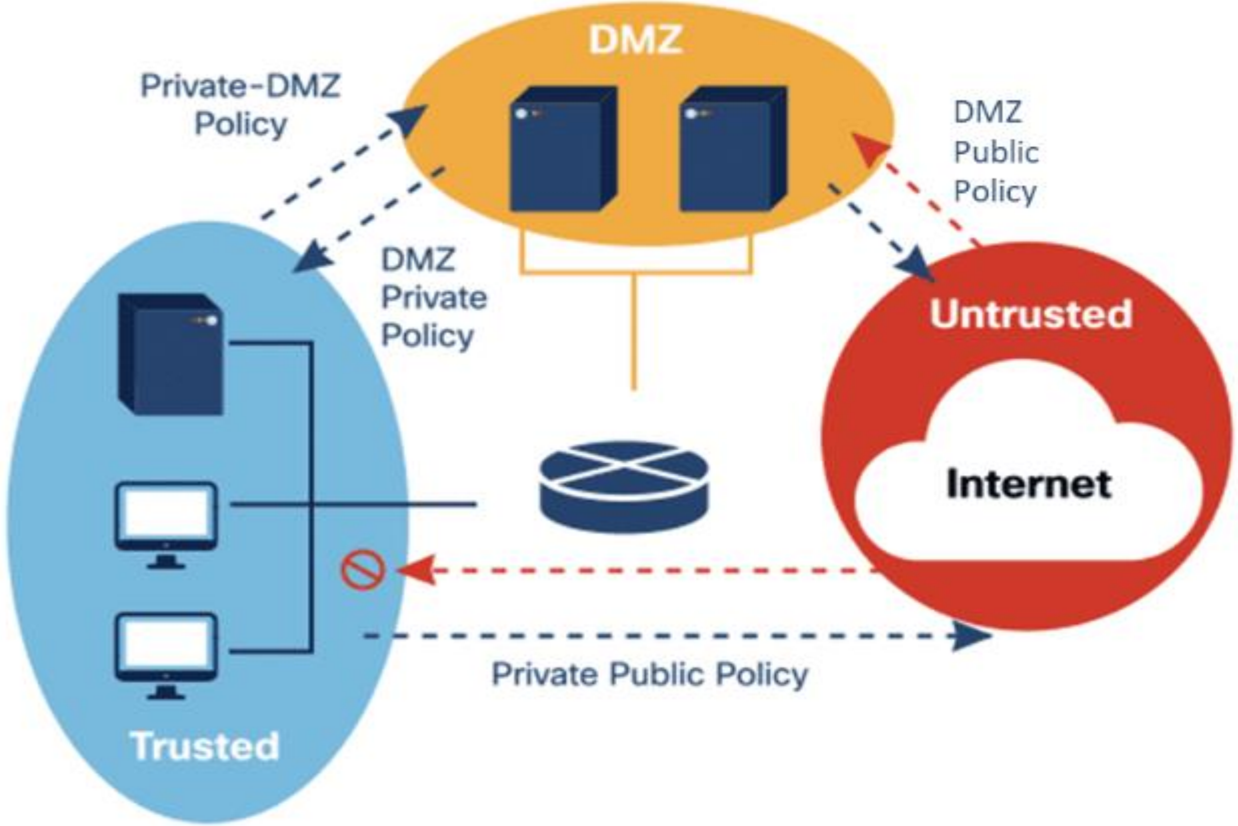
Case 1) Inside and Outside Networks (generic model)



- Perimeter firewall design
  - Deployed at the company edge
  - Primarily about device interfaces permitting or denying traffic based on the source, the destination, and the type of traffic
  - Simple design with two security domains
    - **Private** (inside, trusted or protected) network
      - Allow and inspect traffic flowing out to public
    - **Public** (outside, untrusted or dangerous) network
      - Traffic from the public network to the private network is generally blocked

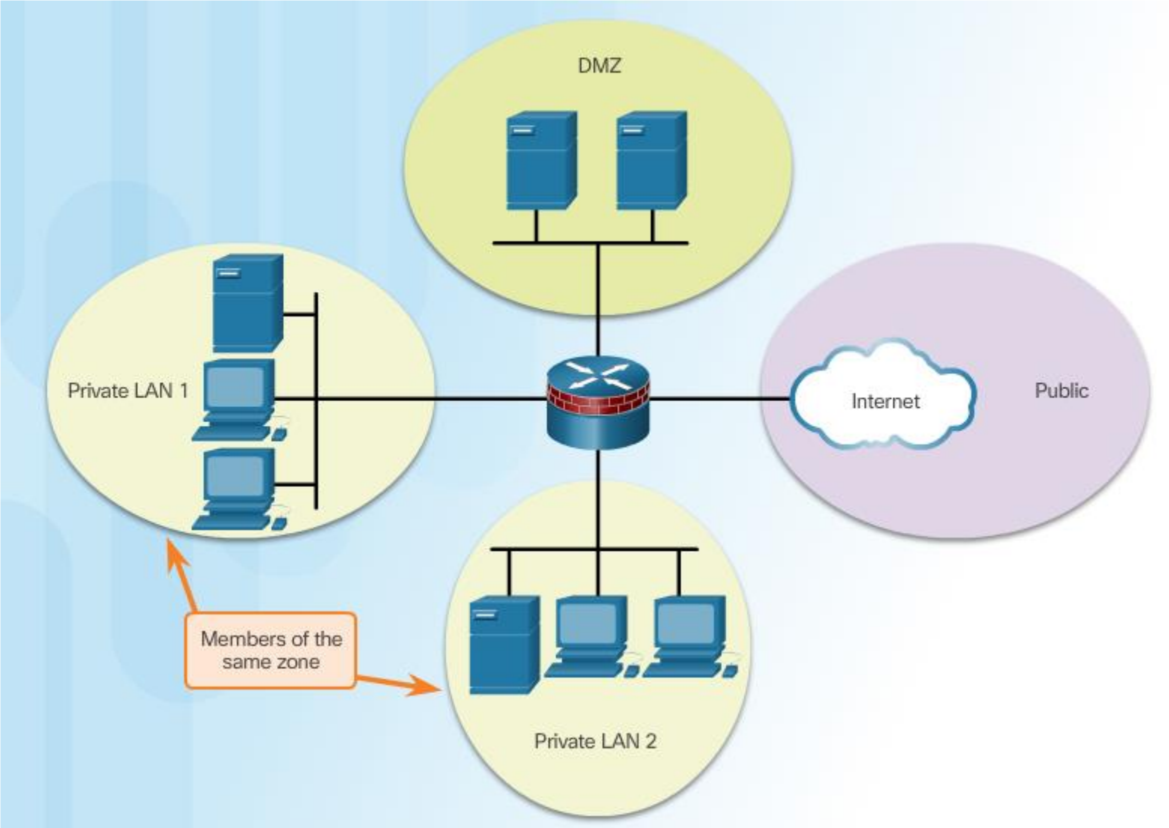
Case 2) Demilitarized Zones (generic model)

- Demilitary Zones (DMZ): security enhancement of previous design
  - DMZ is a network segment with special policy
  - Usually for publicly exposed services
- Zones of risk
  - Inside: risk level low, we trust corporate users
  - DMZ: risk level medium to high
  - Internet/Outside: risk level high
- Latest approach => **Zero Trust model**
  - Do not trust end users even
  - Monitor all network



- DMZ design able to setup FW rules
  - In => Out/DMZ: allow and inspect
  - Out => In: denied, expect of inspected traffic
  - Out <=> DMZ: selectively permitted
  - DMZ => In: usually blocked

Case 3) Zone-Based Firewalls (ZBF)



- ZBF: further security enhancement Introduces **zones**, a logical group of one or more interfaces that have similar functions or features
  - Traffic between interfaces in the same zone => **passes freely**
  - All other zone-to-zone traffic is by default **blocked**, and policy have to be defined
    - Policy: specify what transit (user) traffic is allowed to be initiated (for example, from users on the inside destined to resources on the outside) and what action the firewall should take (inspection, pass, drop)
    - Policy is applied in a single direction
- Notes:
  - Concept of zones may use some other security vendors too
  - ZBF implementation within the Cisco IOS = Zone-Based Policy Firewall (ZPF)

Common security zones include

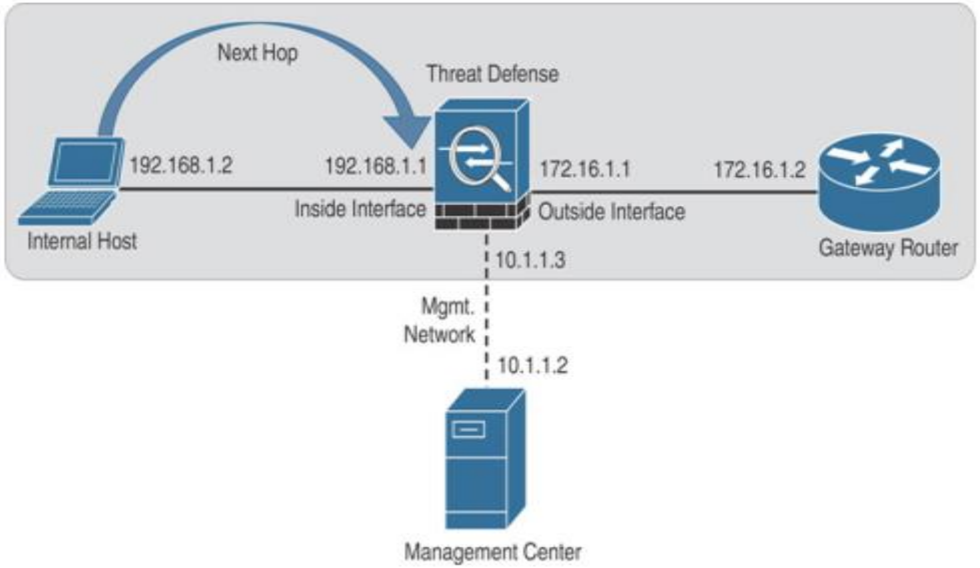
- Zone definition: logical group of one or more interfaces

Inside	Also called the "private" network, this is a zone populated by inside hosts that must be protected from outside hosts. The traffic from the inside is typically permitted to traverse the firewall to the outside with little or no restrictions, whereas traffic returning from the outside that is associated with traffic originating from the inside is permitted to traverse from the untrusted interface to the trusted interface.
Outside	Also called the "public" network, this zone is not to be trusted as it connects to the outside of our network. Traffic originating from the outside is generally blocked entirely or very selectively permitted.
Demilitarized zone (DMZ)	This zone typically connects to servers providing access and services to outside users. For example, hosting a web server, email server, and more.
Self zone	This is a system-defined zone that does not have any interfaces as members. It applies to traffic directed to the router (e.g., SSH, HTTPS, SNMP) or traffic generated by the router (e.g., Syslog, SNMP traps).

Firewall deployment modes

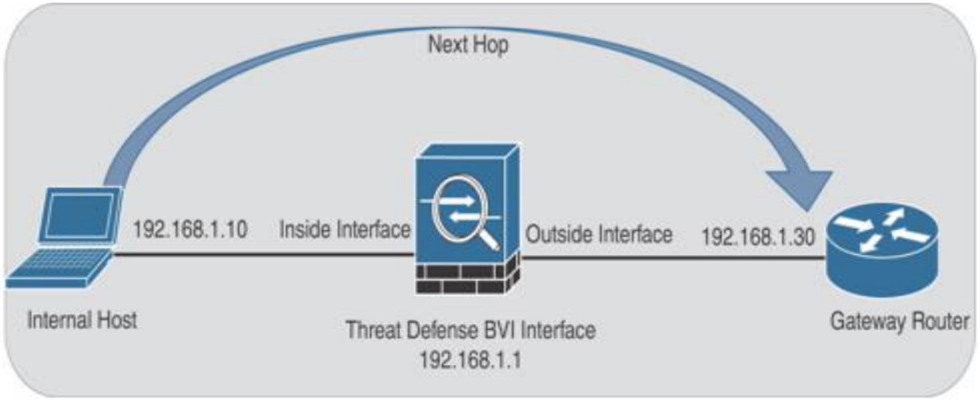
Routed mode

- Operates at L3
  - Is default GW from host point of view
- Separates subnets



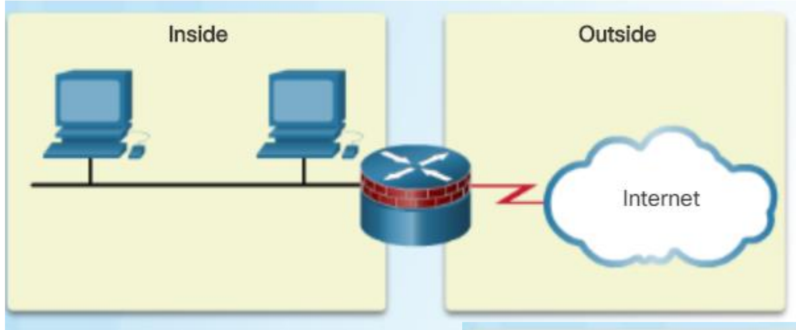
Transparent mode

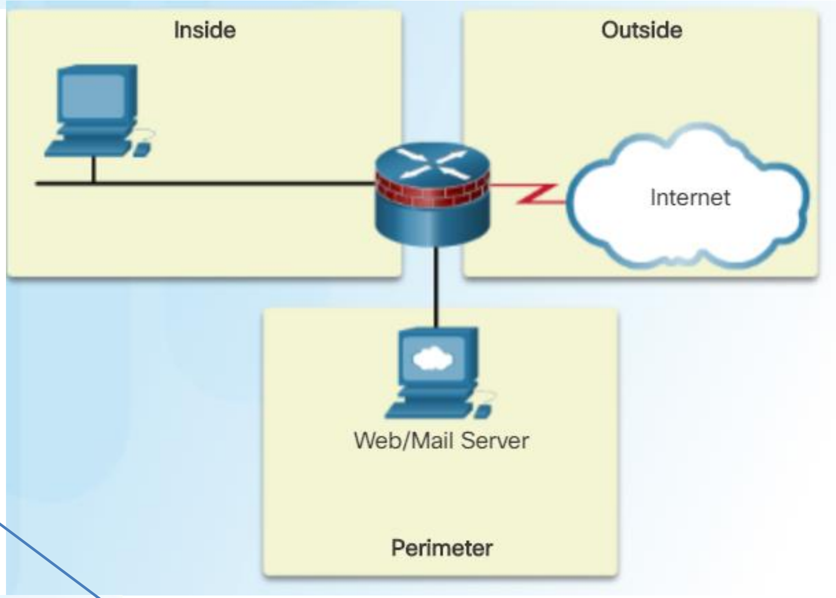
- Operates at L2, network transparent
  - Bridging inside and outside transparently

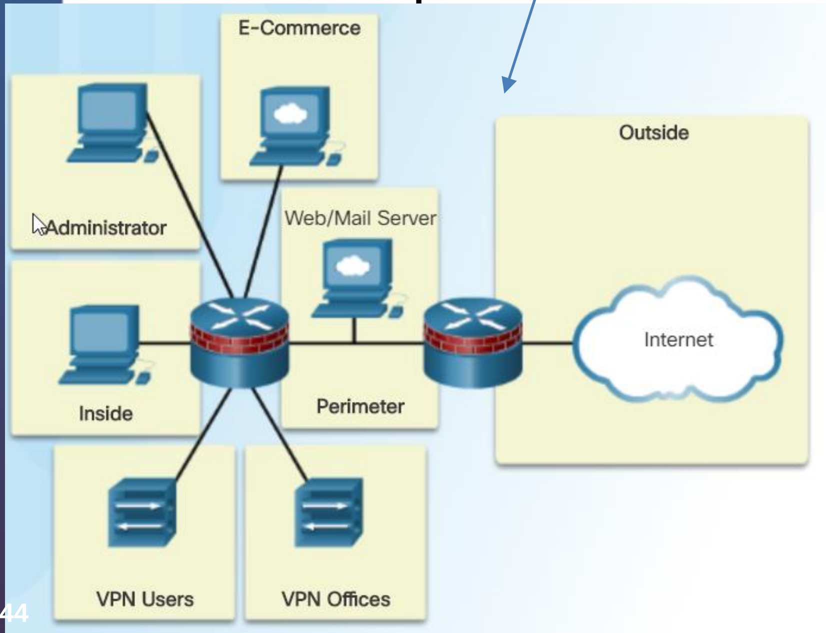


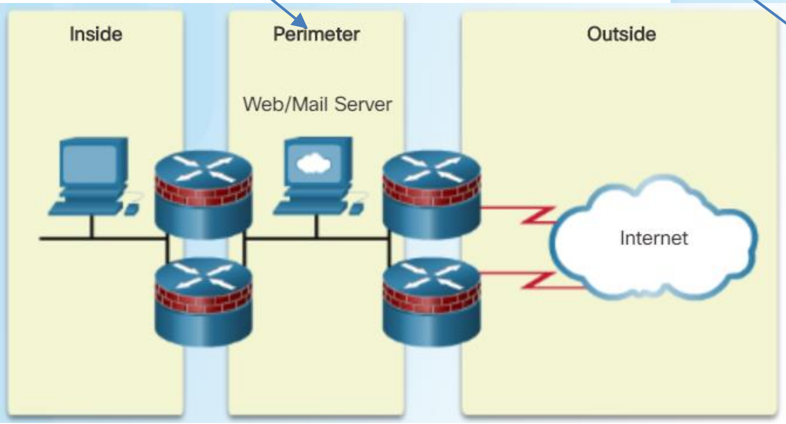
FW design - generic deployment models

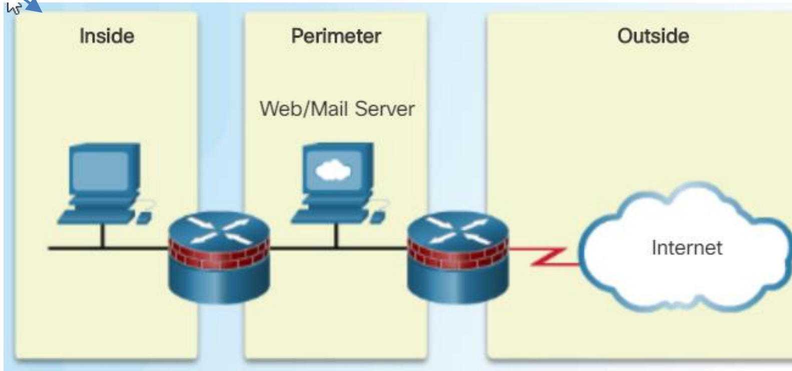
- Common FW designs include:
  - LAN-to-Internet
  - Firewalls between public servers
  - Redundant firewalls
  - Complex firewalls
  - Transparent











Firewalls in network design - Layered Defense

- Considerations for network defense - think on security as different layers
  - Layers provides security depth
  - Use different types of firewalls, policy and policy enforcements combined in layers
  - Firewall can not stop intrusion within inside net
  - *Note: it has hierarchical design, not all layers are required for each network deployments*
- Layers:
  - Network core security
    - Protects against malicious sw, provides anomaly traffic detection, enforces net policies, ensures survivability
  - Perimeter security
    - Secured boundaries between layers
  - Communications security
  - Endpoint security
    - Identity and device policy compliance

- Firewall best practices include:
  - Position firewalls at security boundaries.
  - It is unwise to rely exclusively on a firewall for security.
  - Deny all traffic by default. Permit only services that are needed.
  - Ensure that physical access to the firewall is controlled.
  - Monitor firewall logs.
  - Practice change management for firewall configuration changes. Bad processes lead to incorrect rules.
  - Remember that firewalls primarily protect from technical attacks originating from the outside.

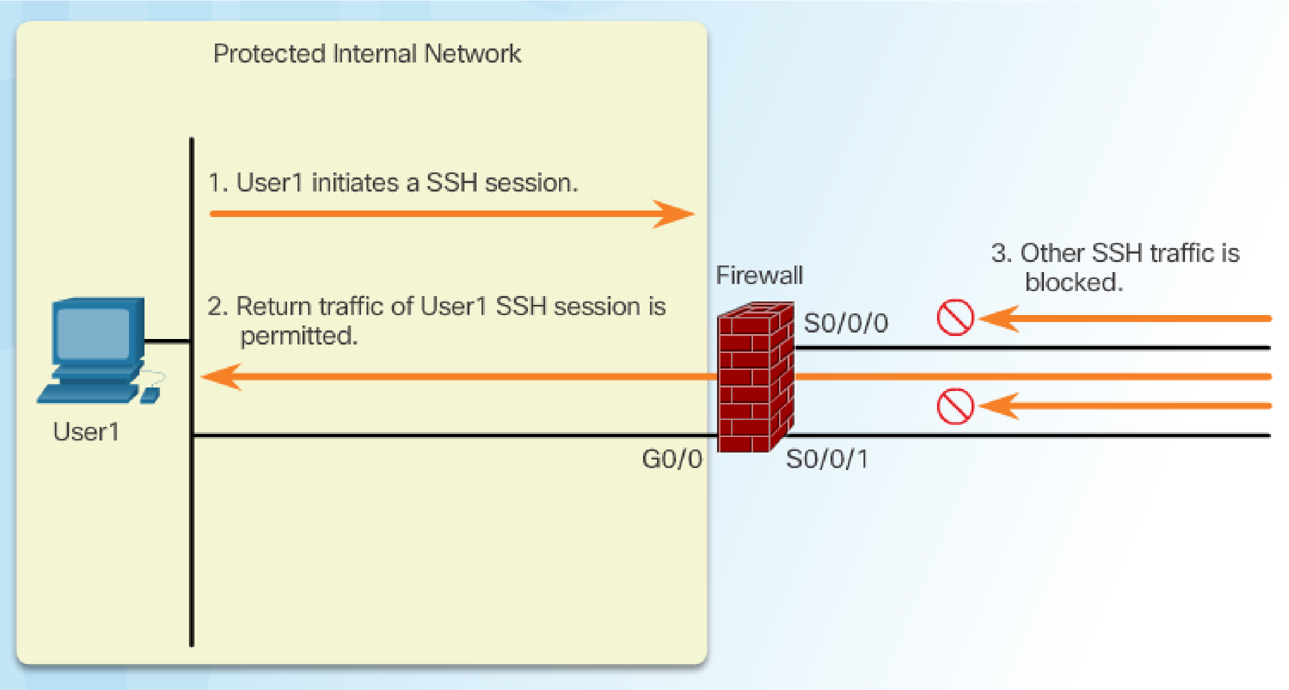
Rule (ACLs) Implementation InConsistency

Rule	Description
Rules that are too promiscuous	These types of rules allow more access than is necessary for the business requirement. Often, a rule may be implemented in an attempt to get a network application working, and the keyword of **any** is allowed for either the addresses or the IP keyword for the entire protocol stack. Unfortunately, if this rule is put in as a temporary test, and the application begins to work, it will be very difficult later in the production environment to narrow the scope of the access and still allow the application to function. Rules that are too promiscuous are significant holes in a security policy.
Redundant rules	ACLs are processed from top to bottom. If a rule is already in place as allowing a specific flow of traffic, a second rule for that does not need to be added to the control lists. Unfortunately, if an ACL is thousands of lines long, or is using object groups that are not understood by the administrator, additional unnecessary entries may be inadvertently added by the administrator. This does not necessarily cause an additional security risk, but it does create rules that are unnecessarily long (or at least longer).
Shadowed rules	A shadowed rule is basically incorrect order placement in the ACL. For example, if you want to deny a specific source IP address from going to a specific web server, and you add the entry for that to an ACL, one would think that that access is now filtered. However, access control entries are added by default to the bottom of an ACL. As a result, if a previous line specifically permits all web traffic to any web server, that entry permits this individual device to go to the specified web server before the new ACL entry is ever considered.
Orphaned rules	This most likely results from a configuration error that is referencing incorrect IP addresses that would never be seen by the firewall. For example, if an ACL intended to filter traffic from inside users includes a source IP address range that does not exist on the inside of the network, that ACL entry will never be matched. Orphaned rules are simply taking up space in the configuration and are never matched.
Incorrectly planned rules	This may result from an error that is made as the business requirements are being translated to the technical and logical controls that the firewall will implement. This may be due to a lack of understanding what protocols (and/or ports) are really used by the devices in the network with the applications in use.
Incorrectly implemented rules	This results from an administrator implementing the incorrect port, protocol, or IP information on the firewall.

Implementing Cisco IOS Firewalls (Cisco IOS Firewall models) / IOS Classic Firewall

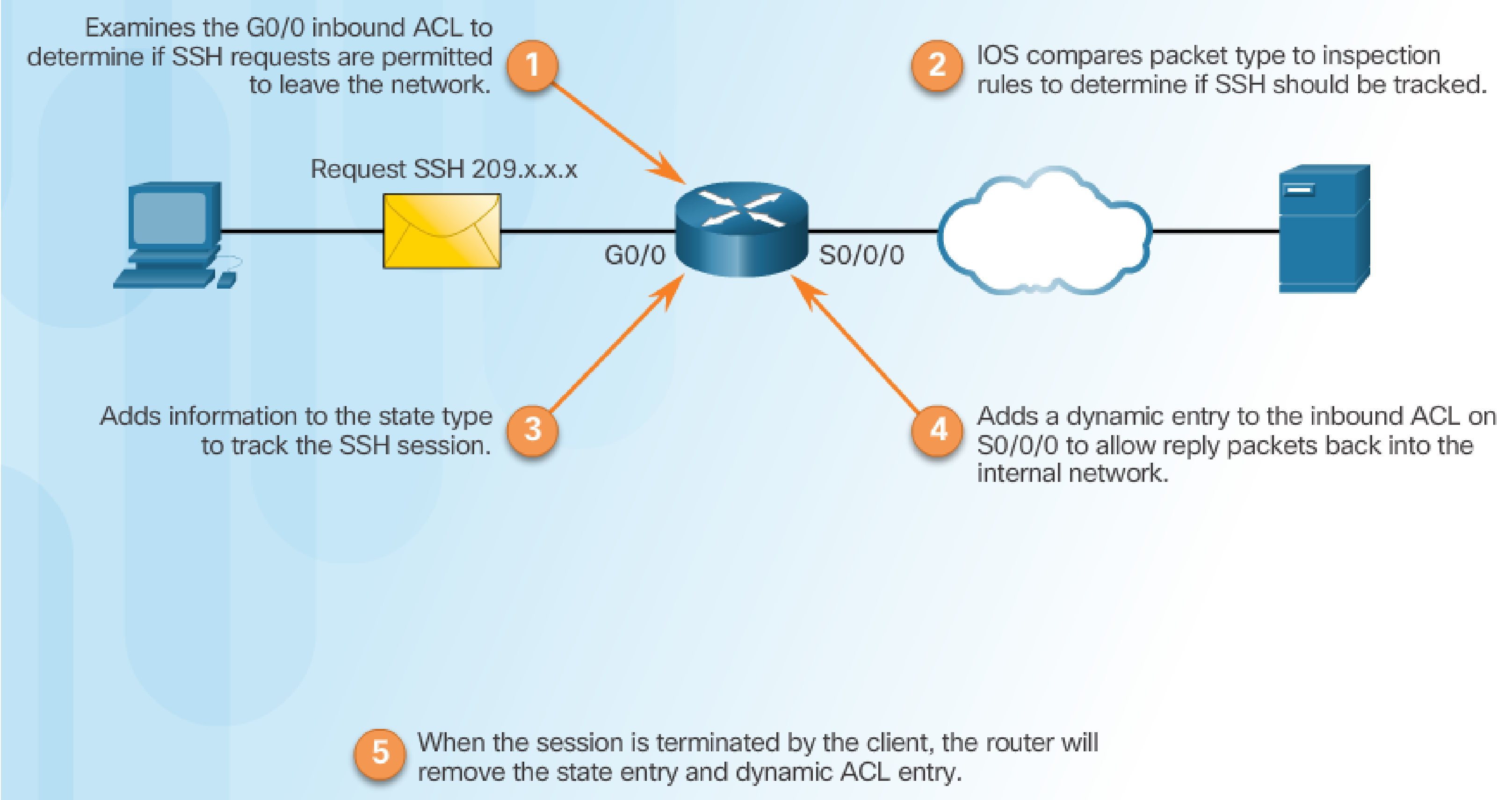
Classic Firewall in IOS

- Cisco IOS Classic Firewall
  - Known before as Context-based access control (CBAC) (renamed then to SPI)
  - Provides Statefull firewall feature of Cisco IOSs prior the version 12.0
  - Functions:
    - Traffic filtering
    - L7 traffic inspection (Statefull Packet Inspection feature)
    - Intrusion detection
    - Generation of alerts and audits
- SPI works only for protocols defined by admin and supported by IOS
  - Peer2Peer, messaging, dns, http, ftp, esmtp, h323, skinny …
  - Supports protocol that use multichannel (FTP)



- Detects only attacks those go through the firewall
- All traffic passing through that interface received the same inspection policy (low granularity)
- Still maintained but not enhanced
  - Further Cisco IOS development focuses on Zone-Based Policy Firewall (ZPF)

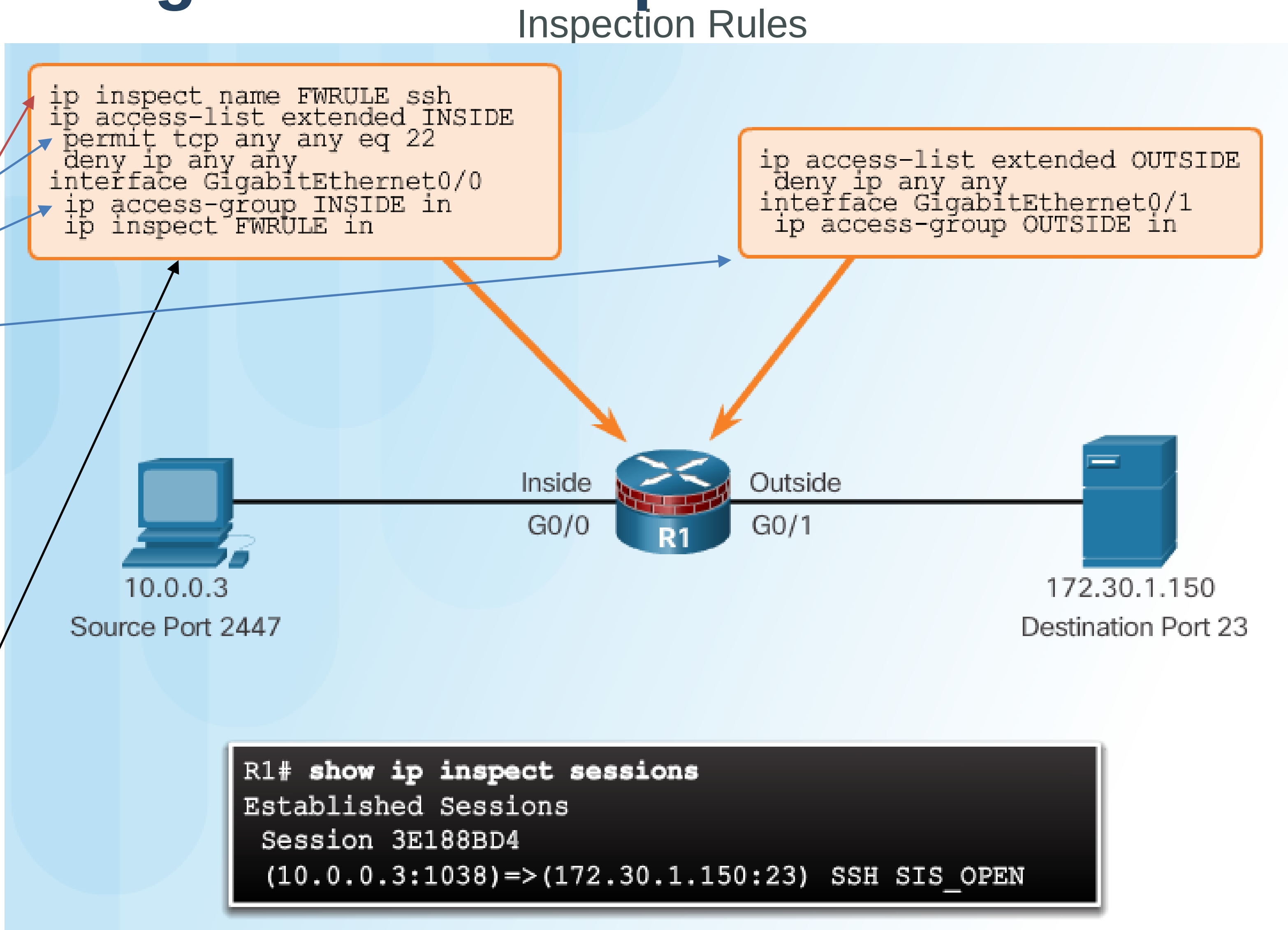
Classic Firewall Operation



- The functionality expects for each protocol to have defined ACL and inspection rules
- Then
  - If ACL action is permit and inspection rule for a protocol (e.g. SSH) is defined
    - The state record is created
    - Dynamic temporary ACL ACE in backward direction is created to allow packets of the session return
    - Once the session terminate, dynamic ACL ACE is deleted
  - If ACL is permit and no inspection is defined
    - Traffic is allowed, however admin have to define manually how packets will return back (stateless behavior)
  - If ACL ACE action is drop, packets are denied

Classic Firewall Configuration example

1. Choose the internal and external interfaces
2. Configure ACLs for each interface
3. Define inspection rules.
4. Apply an inspection rule to an interface.



Classic FW configuration - CLI

```
! 1) Configure Access Lists
! access-list access-list-number {deny | permit} protocol source
source-wildcard [operator [port]] destination

! 2) Configure Inspection Rules
ip inspect name INSPECTION-NAME PROTOCOL
! For example > Router(config)# ip inspect name firewall tcp

! 3) Apply Access Lists and Inspection Rules to Interfaces
interface TYPE NUMBER
ip access-group {ACCESS-LIST-NUMBER | ACCESS-LIST-NAME}{in | out}
ip inspect INSPECTION-NAME {in | out}
```

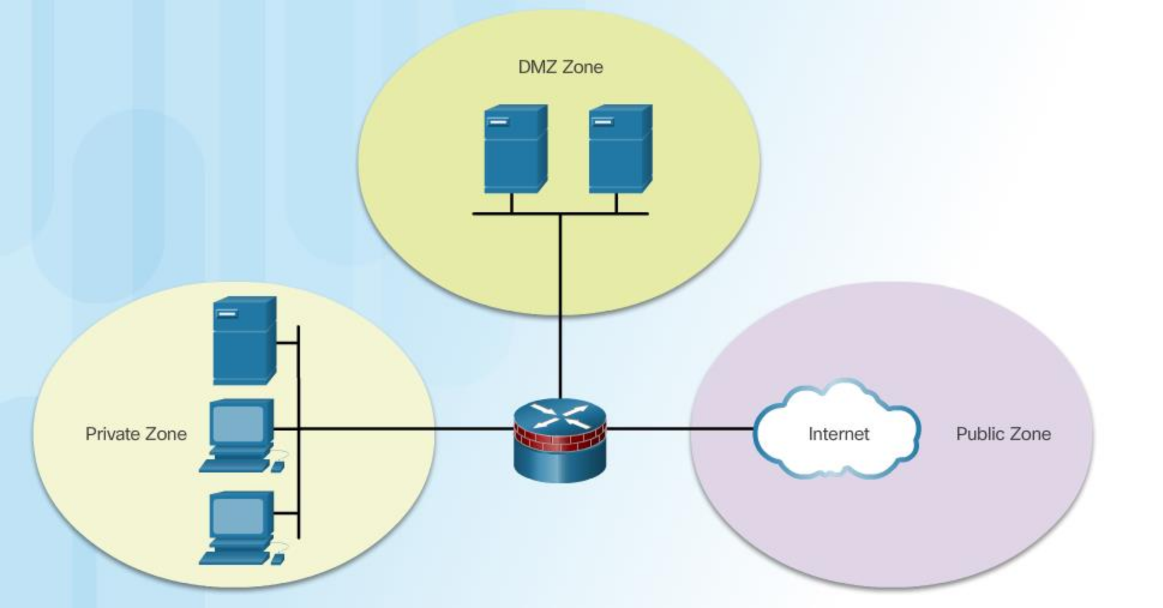
Implementing Cisco IOS Firewalls (Cisco IOS Firewall models) / Zone-Based Policy Firewalls

Upon completion of this section, you should be able to:

- Explain how Zone-Based Policy Firewalls are used to help secure a network.
- Explain the operation of a Zone-Based Policy Firewall.
- Configure a Zone-Based Policy Firewall with CLI.

Benefits of ZPF

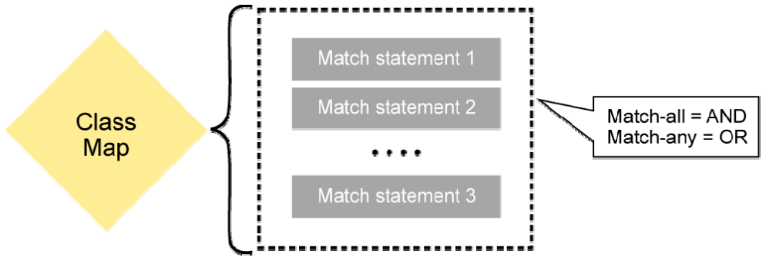
- ZPF
  - The most advanced method of a stateful firewall that is available on Cisco IOS routers
  - Introduces a new firewall model
    - Security zones with defined polices, not ACLs
  - Policies are applied to traffic moving between zones
- Other features
    - Stateful packet inspection (Packet filtering. …)
    - Application inspection
    - URL filtering
    - Transparent firewall (implementation method).
    - Support for virtual routing and forwarding (VRF)
- Both models (classic and ZBF/ZPF)
  - Can be enabled concurrently
  - Cannot be combined on a single interface



- Why to use ZPF?
  - Structured and ease of use
  - Not dependent on ACLs
  - Router security posture is to block unless explicitly allowed
  - Policies are easy to read and troubleshoot with C3PL
    - CPL style of configuration (class-map, policy-map, service-map with if-then statements)
      - If traffic matches class, then perform action
        - Else, if …..
  - One policy affects any given traffic, instead of needing multiple ACLs and inspection actions

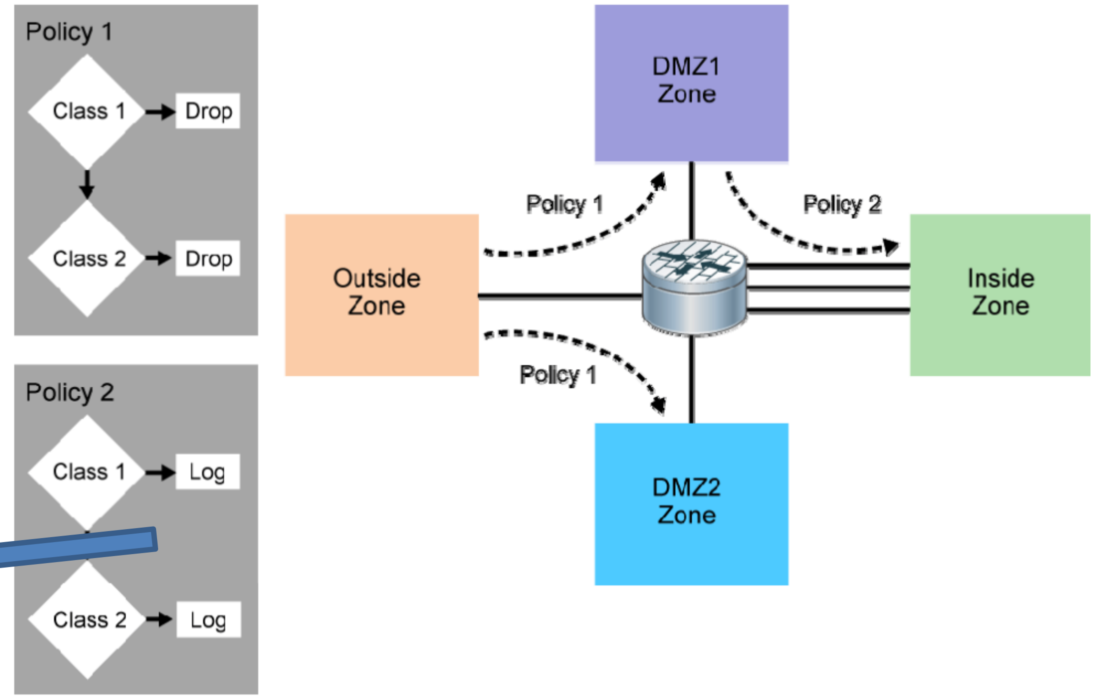
Cisco Common Classification Policy Language - principle

- Class maps



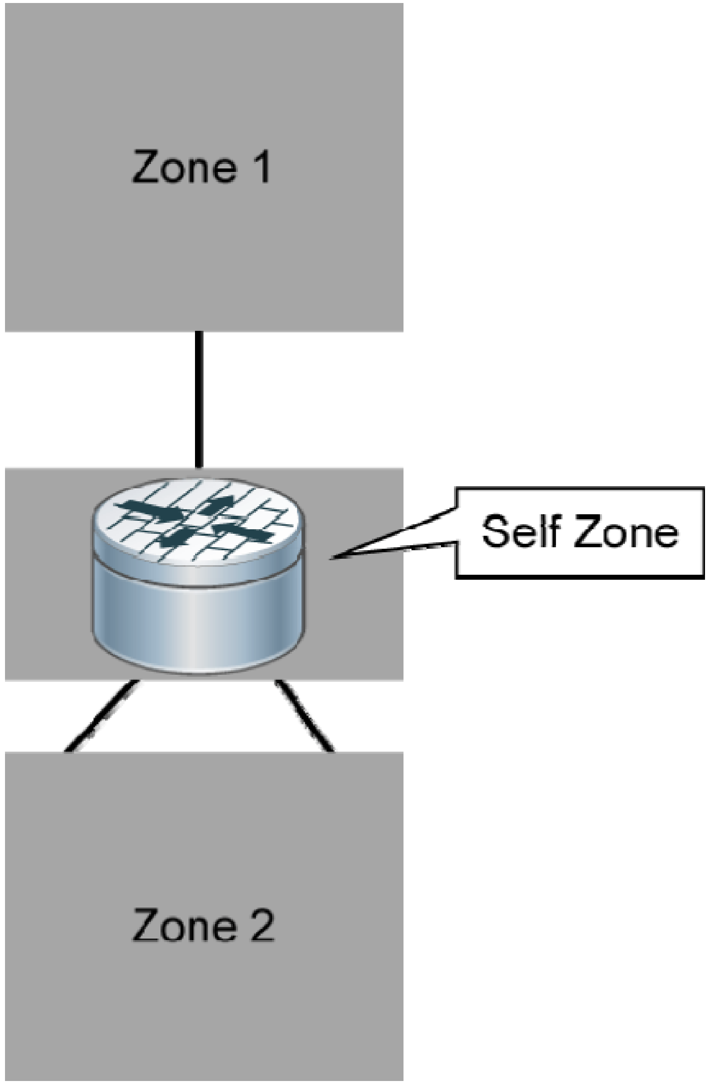
  - L3/L4 traffic analysis and classification based on
    - ACL groups
    - Protocol identification (as IOS recognize)
    - Class map with match criterias
      - Match-any (OR) or match all (AND) statements
- Policy
  - Actions applied to matched traffic
    - Inspect
    - Pass
    - Drop
    - Log

- Two class example



ZPF terminology

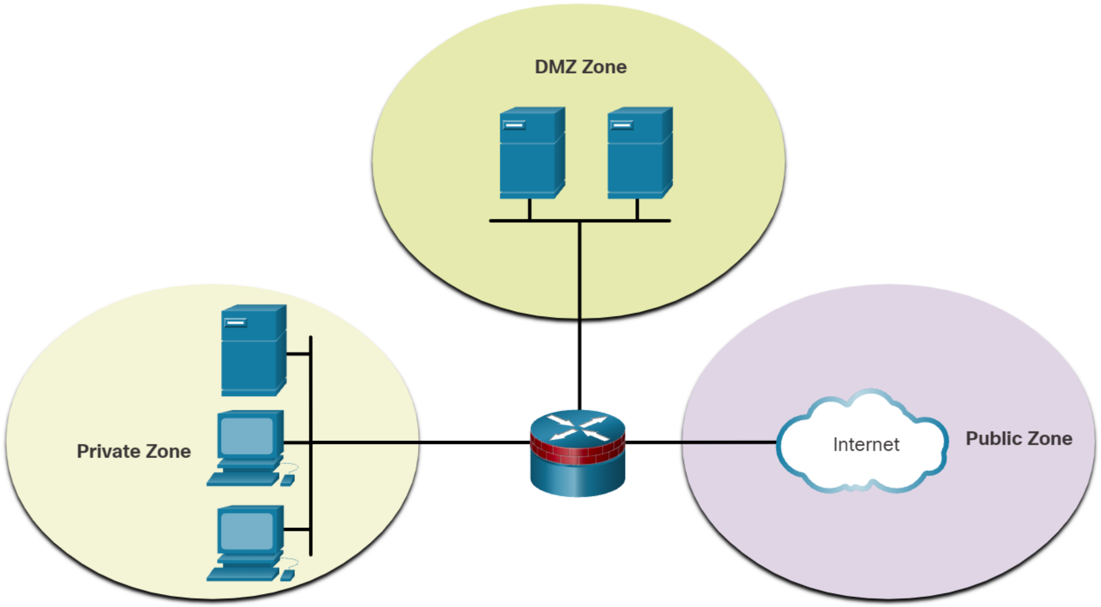
- Zone
  - A zone is a group of interfaces that have similar functions or features
  - Zones provide a way to specify where a Cisco firewall is applied
  - An interface can belong to only one zone
- Security Zones
  - A security zone is a group of interfaces to which a policy can be applied
- Zone Pairs
  - A zone pair allows to specify a **unidirectional firewall policy** between two security zones
  - Need to specify source, destination or self zones



ZPF Actions (policies)

- **Three possible actions – ZPF policies**
  - **Inspect**
    - Perform stateful packet inspections.
  - **Pass**
    - Analogous to a permit statement in ACL.
    - The pass action does not track the state of connections or sessions within the traffic
  - **Drop**
    - Analogous to a deny statement in ACL
    - Log option is available to log the rejected packets
  - **Log**
    - Log the packet

ZPF Rules

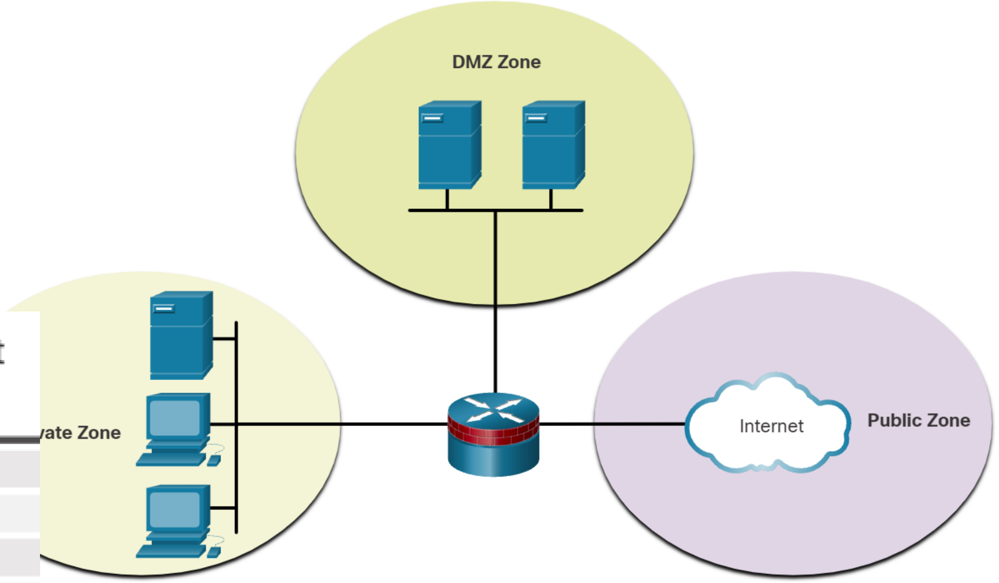


- Depend on whether or not the ingress and egress interfaces are members of the same zone
  - **If neither interface** is a zone member, then the resulting action is to pass the traffic.
  - **If both interfaces are members** of the same zone, then the resulting action is to pass the traffic.
  - **If one interface is a zone member, but the other is not**, then the resulting action is to drop the traffic regardless of whether a zone-pair exists.
  - If both interfaces belong to the same zone-pair and a policy exists, then the resulting action is inspect, allow, or drop as defined by the policy.

Rules for Transit Traffic

- Traffic going through router interfaces

Source Interface Member of Zone?	Destination Interface Member of Zone?	Zone-Pair Exists?	Policy Exists?	Result
NO	NO	N/A	N/A	PASS
YES	NO	N/A	N/A	DROP
NO	YES	N/A	N/A	DROP
YES (private)	YES (private)	N/A	N/A	PASS
YES (private)	YES (public)	NO	N/A	DROP
YES (private)	YES (public)	YES	NO	PASS
YES (private)	YES (public)	YES	YES	INSPECT

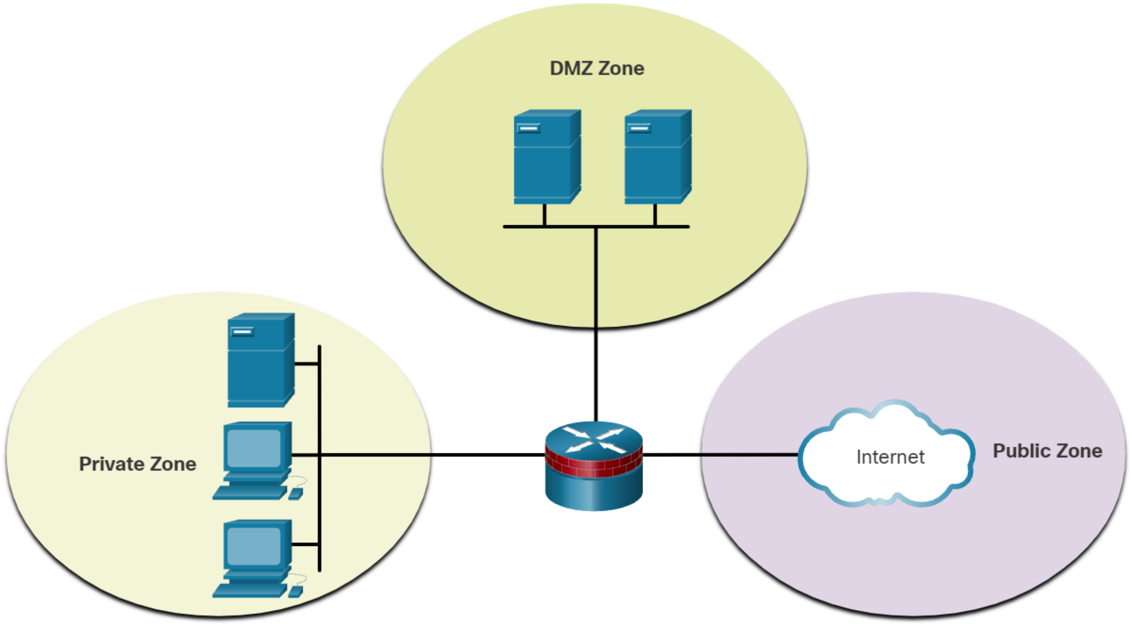


Rules for Traffic to the Self Zone

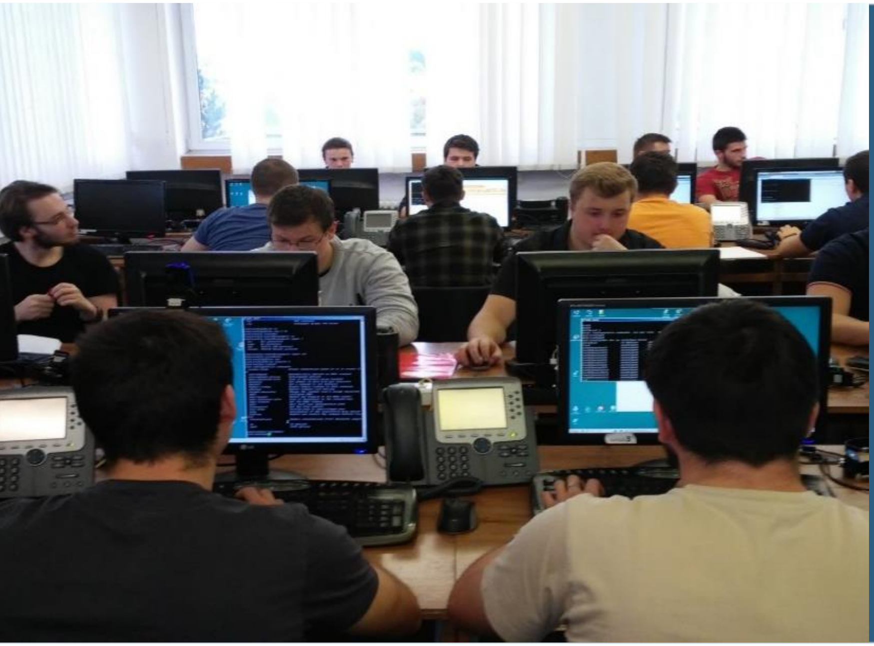
- Self-Zone: router interfaces and all theirs IP addresses

Source Interface Member of Zone?	Destination Interface Member of Zone?	Zone-Pair Exists?	Policy Exists?	Result
YES (self-zone)	YES	NO	N/A	PASS
YES (self-zone)	YES	YES	NO	PASS
YES (self-zone)	YES	YES	YES	INSPECT
YES	YES (self-zone)	NO	N/A	PASS
YES	YES (self-zone)	YES	NO	PASS
YES	YES (self-zone)	YES	YES	INSPECT

ZPF Operating Principle

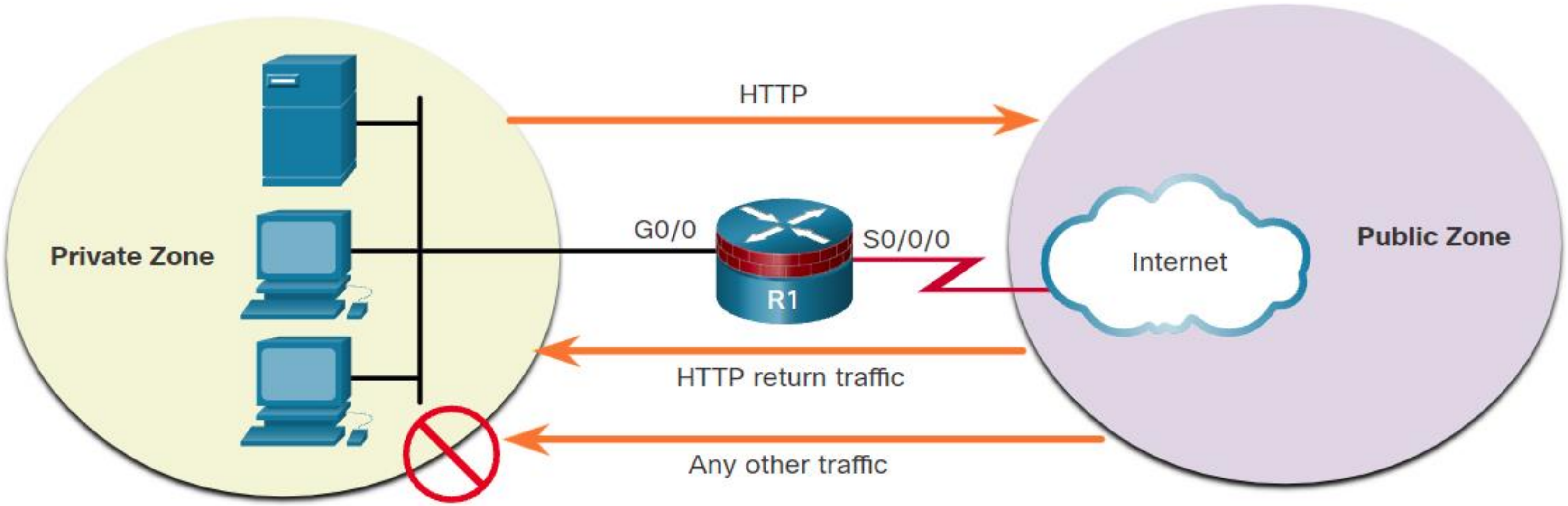


- IOS zone-based firewall
  - Use the concept of security **zones** and **zone pairs**
  - A **zone pair** is directional
    - We are specifying a source zone and a destination zone
  - **Interfaces** are assigned **to zones**.
  - **Firewall policies** are assigned **to zone pairs**.
    - The policy assigned to the zone pair controls traffic that enters the router through the source zone and leaves the router through the destination zone.
    - There is also a system defined zone that is named the self zone.
    - The self zone comprises the IP addresses of the router itself
  - **Default** inter zone policy **is Deny (Drop)**



Configuring a ZPF

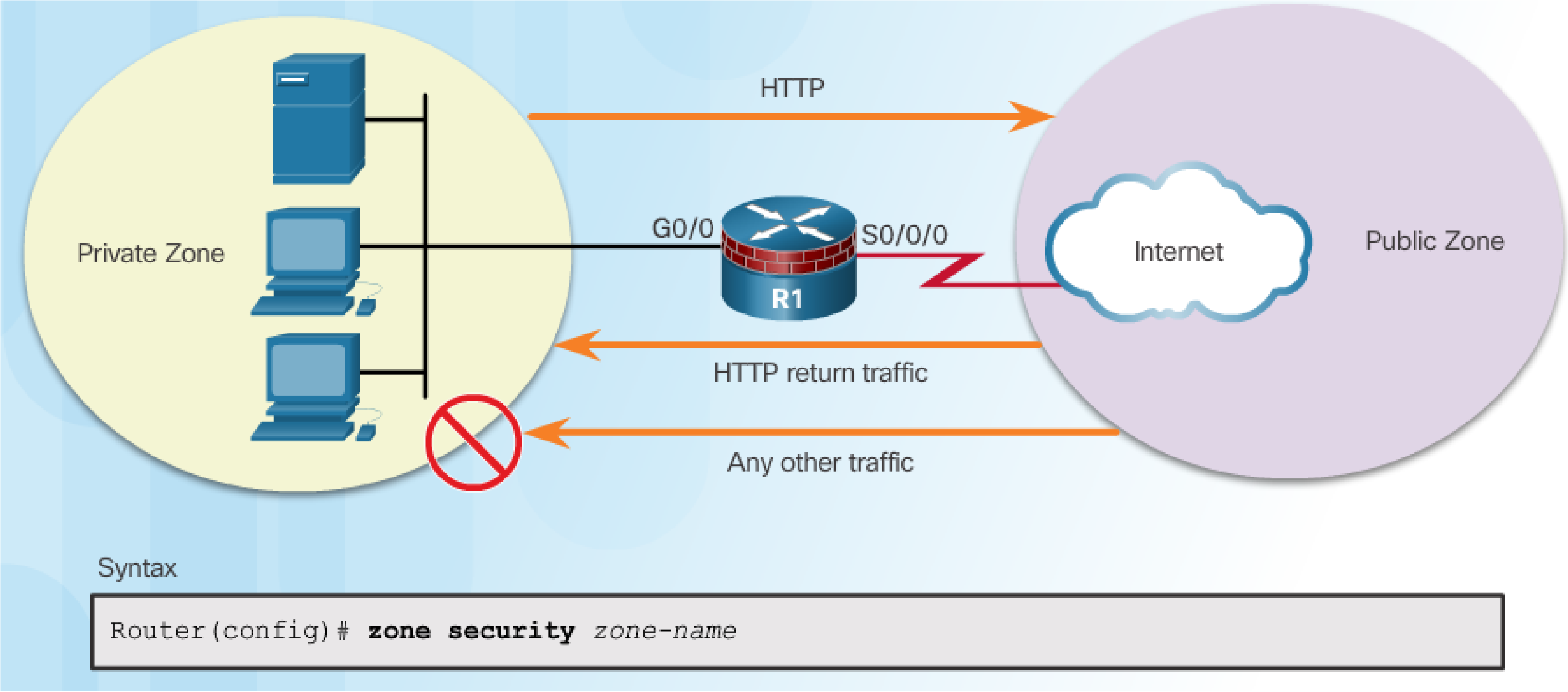
Configure a ZPF



Step 1: Create/define zones.
Step 2: Identify traffic with a class-map.
Step 3: Define an action with a policy-map.
Step 4: Identify a zone pair and match it to a policy-map.
Step 5: Assign interfaces to the appropriate zones.

Step 1: Create the Zones

- Find answer on questions:
  - What interfaces should be included in the zones?
    - Example: two
  - What will be the name for each zone?
    - Public/Private
    - Inside/Outside…
  - What traffic is necessary between the zones and in which direction?
    - Example: allow HTTP/DNS to go out




```
! Create the security zones, they can be named whatever you
! want to name them. In this example:
R1(config)# zone security PRIVATE
R1(config-sec-zone)# exit
R1(config)# zone security PUBLIC
R1(config-sec-zone)# exit
```

Step 2: Identify Interesting Traffic using class-map

1. Create a Class-map:

Keyword for ZPF: inspect

```
Router(config)# class-map type inspect [match-any | match-all] class-map-name
```

Parameter	Description
match-any	Packets must meet one of the match criteria to be considered a member of the class.
match-all	Packets must meet all of the match criteria to be considered a member of the class.
class-map-name	Name of the class-map used to configure the policy for the class in the policy-map.

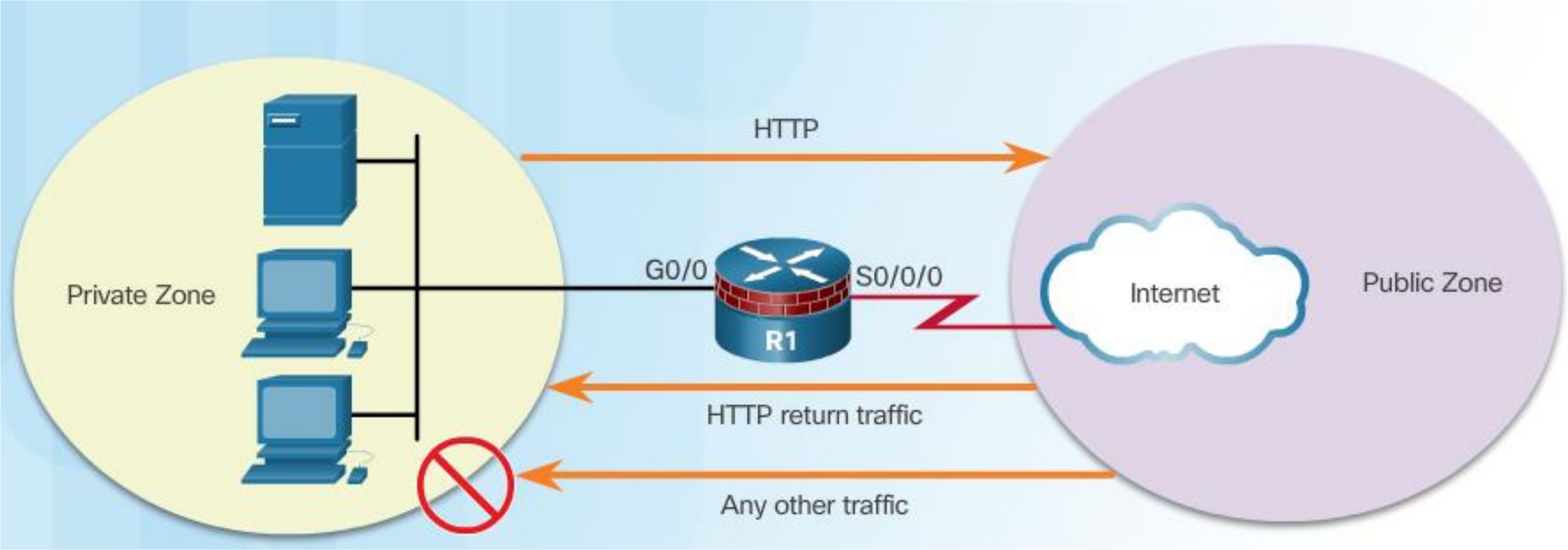
2. Specify matching criteria:

```
Router(config-cmap)# match access-group {acl-# | acl-name }
Router(config-cmap)# match protocol protocol-name
Router(config-cmap)# match class-map class-map-name
```

Parameter	Description
match access-group	Configures the match criteria for a class-map based on the specified ACL number or name.
match protocol	Configures the match criteria for a class-map based on the specified protocol.
match class-map	Uses another class-map to identify traffic.

Example `class-map` configuration

Step 2: Identify Traffic (Cont.)




```
! The class map "classifies" or "identifies" the traffic
! In this example, this class map will match on either HTTP and DNS traffic

R1(config)# class-map type inspect match-any HTTP-TRAFFIC
R1(config-cmap)# match protocol http
R1(config-cmap)# match protocol https
R1(config-cmap)# match protocol dns
R1(config-cmap)# exit
```

Step 3: Define an Action using policy-map

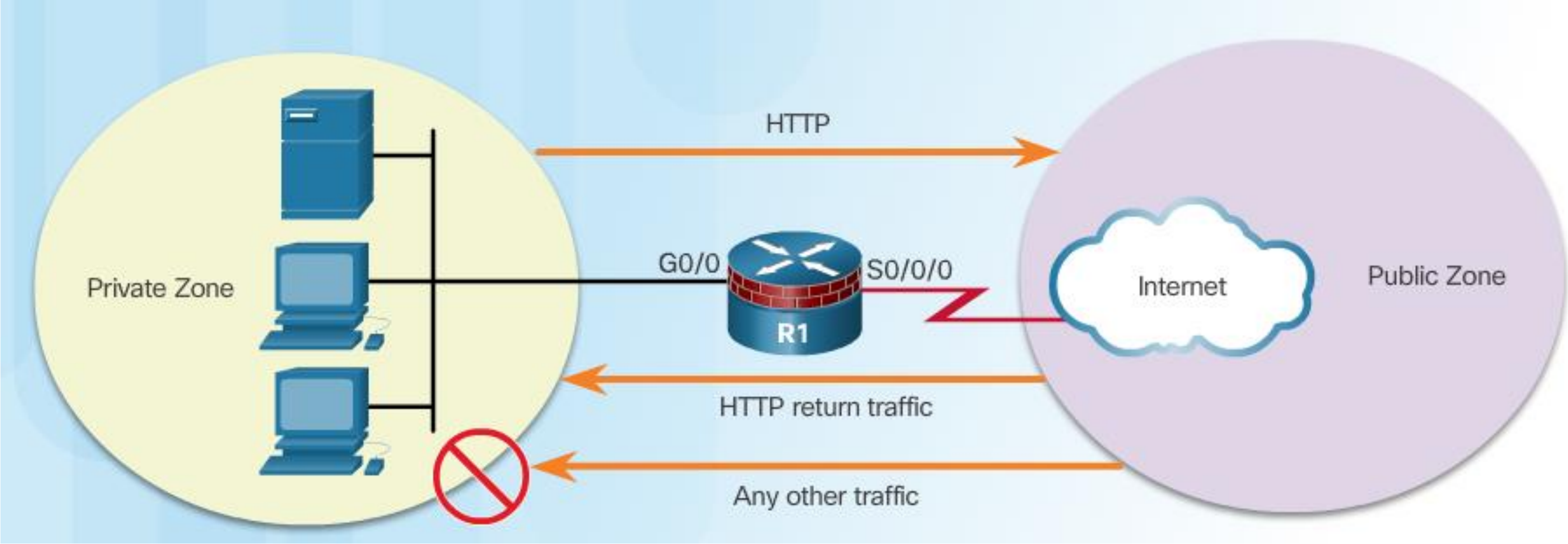
- Use policy-map to define what action would be taken:
  - Inspect
    - Perform statefull-based traffic control, i.e. permit return traffic for initiated sessions
  - Drop
    - Discard unwanted traffic
      - Default action similar to deny any
    - ICMP is not generated
  - Pass
    - Stateless forward traffic from one zone to another
      - i.e. in one direction
    - Does not create state table records

```
Router(config)# policy-map type inspect policy-map-name
Router(config-pmap)# class type inspect class-map-name
Router(config-pmap-c)# { inspect | drop | pass }
```

Parameter	Description
inspect	An action that offers statebased traffic control. The router maintains session information for TCP and UDP and permits return traffic.
drop	Discards unwanted traffic
pass	A stateless action the allows the router to forward traffic from one zone to another

Example `policy-map` configuration

Step 3: Define an Action (Continue)




```
! The policy map calls on a specific class map that it wants to use
! to identify which traffic the policy applies to, and then specifies the
! policy action. In this example, it is to inspect the traffic
! Note: It may call more different class-maps
R1(config)# policy-map type inspect PRIV-TO-PUB-POLICY
R1(config-pmap)# class type inspect HTTP-TRAFFIC
R1(config-pmap-c)# inspect
R1(config-pmap-c)# exit
R1(config-pmap)# exit
```

Step 4: Identify a Zone-Pair and Match to a Policy

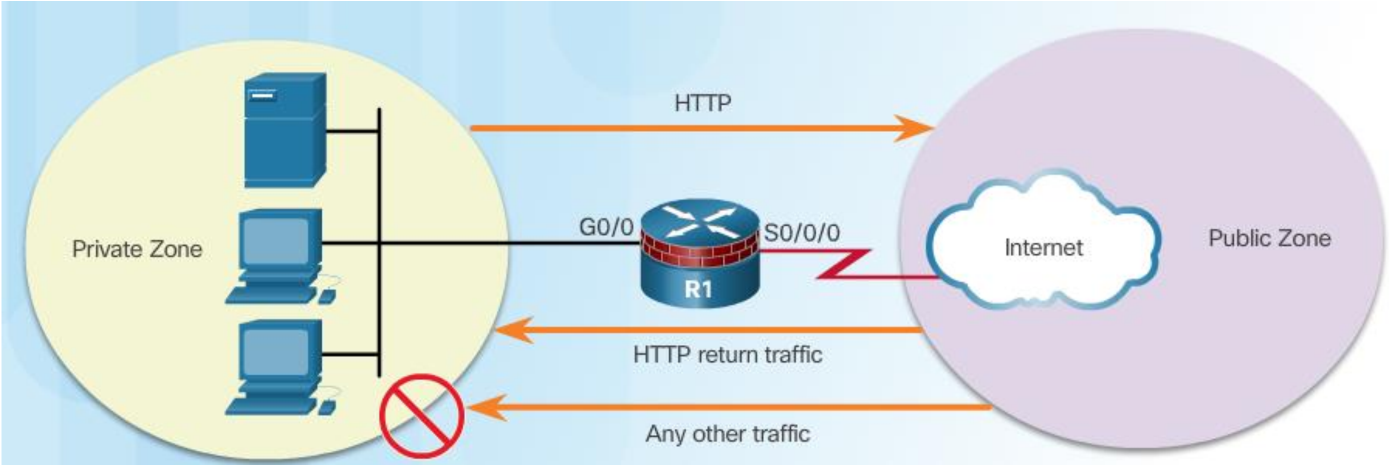
- 1) Create a zone pair
- 2) Attach policy map and associate actions

```
Router(config)# zone-pair security zone-pair-name source {source-zone-name | self
} destination {destination-zone-name | self }
Router(config-sec-zone-pair)# service-policy type inspect policy-map-name
```

Parameter	Description
source source-zone-name	Specifies the name of the zone from which traffic is originating.
destination destination-zone-name	Specifies the name of the zone to which traffic is destined.
self	Specifies the system-defined zone. Indicates whether traffic will be going to or from the router itself.

Step 4: Identify a Zone-Pair and Match to a Policy (Cont.)

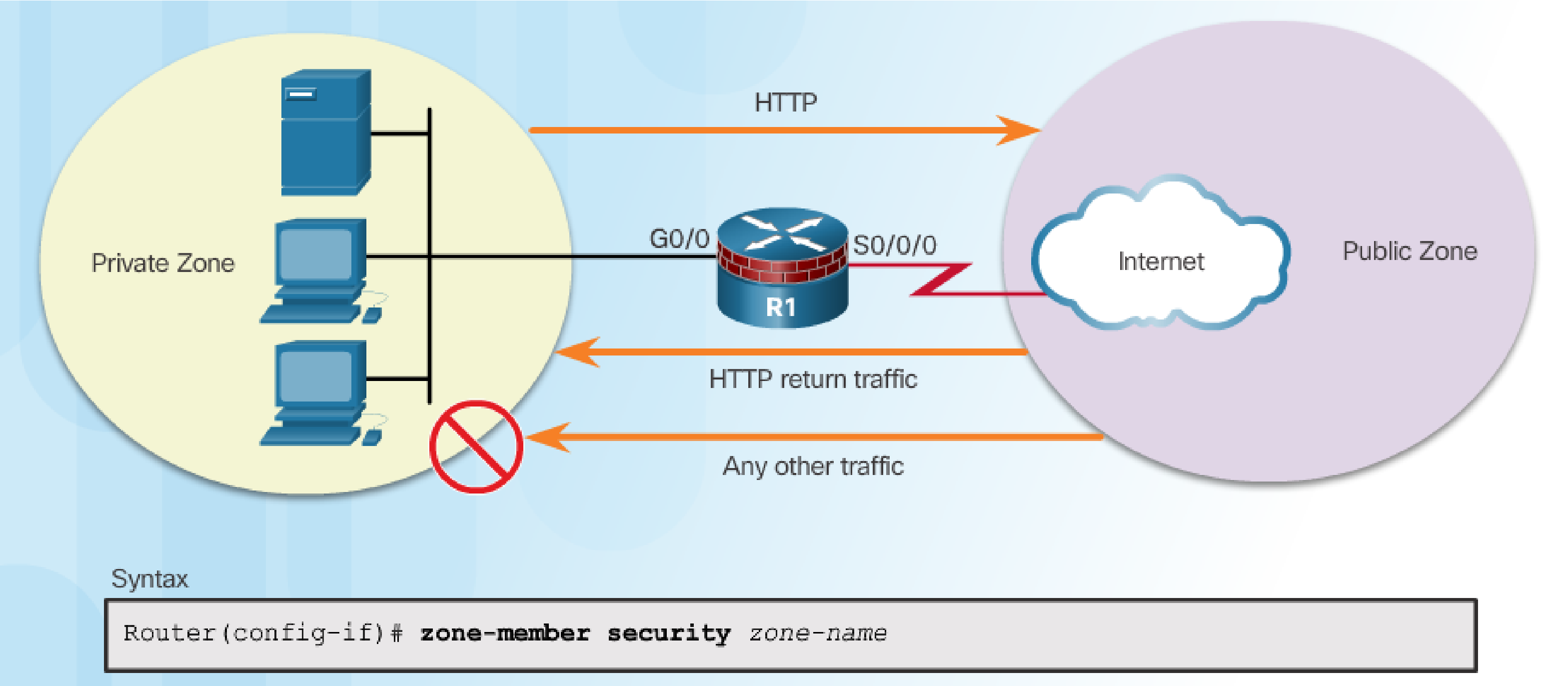
- 1) Create a zone pair
  - i.e. PRIV-PUB
  - Define source PRIVATE
    - Traffic coming from
  - Destination PUBLIC
    - Traffic going to
- 2) Attach policy map and associated actions
  - i.e. PRIV-TO-PUB-POLICY




```
! Create the zone-pair, specifying the zones and the direction (from where
! to where)
R1(config-sec-zone)# zone-pair security PRIV-PUB source PRIVATE destination PUBLIC
! Use the service-policy command in zone-pair configuration mode to apply
! the policy map you want to use for traffic that matches this zone-pair
R1(config-sec-zone-pair)# service-policy type inspect PRIV-TO-PUB-POLICY
R1(config-sec-zone-pair)# exit
```

Step 5: Assign Zones to Interfaces

- Command immediately apply the service-policy
  - Note: If not defined => DROP all packets
- Example
  - Allows HTTP/HTTPS/DNS from private to public
  - and only associated traffic allows back




```
! Configure the interfaces, so they become members of the
! respective zones
R1(config)# interface GigabitEthernet0/0
R1(config-if)# description Belongs to PRIVATE zone
R1(config-if)# zone-member security PRIVATE
R1(config-if)# exit
R1(config)# interface Serial0/0/0
R1(config-if)# description Belongs to PUBLIC zone
R1(config-if)# zone-member security PUBLIC
R1(config-if)# exit
```

Verification

Verification commands:

- show run | begin class-map
- show policy-map type inspect zone-pair sessions
- show class-map type inspect
- show zone security
- show zone-pair security
- show policy-map type inspect

show policy-map type inspect zone-pair sessions

```
R1# show policy-map type inspect zone-
pair sessions

policy exists on zp PRIV-PUB
  Zone-pair: PRIV-PUB

  Service-policy inspect : PRIV-TO-PUB-
POLICY

    Class-map: HTTP-TRAFFIC (match-any)
      Match: protocol http
        12 packets, 384 bytes
        30 second rate 0 bps
      Match: protocol https
        5 packets, 160 bytes
        30 second rate 0 bps
      Match: protocol dns
        0 packets, 0 bytes
        30 second rate 0 bps

! … continue … =>
```

```
! … continue … =>

Inspect

  Number of Established Sessions = 1
  Established Sessions
   Session 2204E220
(192.168.1.3:1049)=>(10.1.1.2:443)
https:tcp
  SIS_OPEN/TCP_CLOSEWAIT
         Created 00:00:14, Last heard
00:00:11
         Bytes sent
(initiator:responder) [821:1431]


    Class-map: class-default (match-any)
      Match: any
      Drop
        4 packets, 160 bytes
R1#
```

show class-map type inspect

```
R1# show class-map type inspect
Class Map type inspect match-any
HTTP-TRAFFIC (id 1)
   Match protocol http
   Match protocol https
   Match protocol dns

R1# show zone security
zone self
Description: System Defined Zone

zone PRIVATE
 Member Interfaces:
 GigabitEthernet0/0


! … continue … =>
```

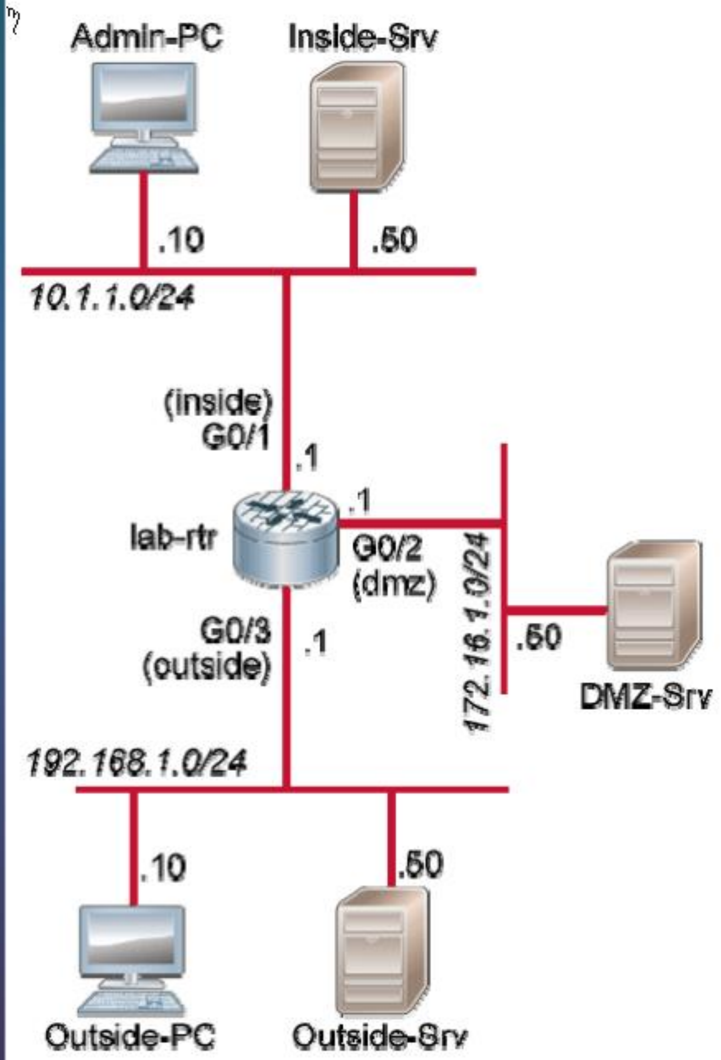
```
! … continue … =>


zone PUBLIC
 Member Interfaces:
 Serial0/0/0

R1# show zone-pair security
Zone-pair name PRIV-PUB
    Source-Zone PRIVATE
Destination-Zone PUBLIC
    service-policy PRIV-TO-PUB-
POLICY

R1# show policy-map type inspect
Policy Map type inspect PRIV-TO-PUB-
POLICY
    Class HTTP-TRAFFIC
      Inspect
    Class class-default
      Drop
```

A ZPF config example with ACL classification

```
class-map type inspect match-all all-private
match access-group 101
class-map type inspect match-all private-ftp
  match  protocol ftp
  match  access-group 101
class-map type inspect match-any netbios
  match  protocol msrpc
  match  protocol netbios-dgm
match protocol netbios-ns
  match  protocol netbios-ssn
class-map type inspect match-all private-netbios
match class-map netbios
  match  access-group 101
class-map type inspect match-all private-ssh
match protocol ssh
  match  access-group 101
class-map type inspect match-all private-http
match protocol http
  match  access-group 101
!
!
```

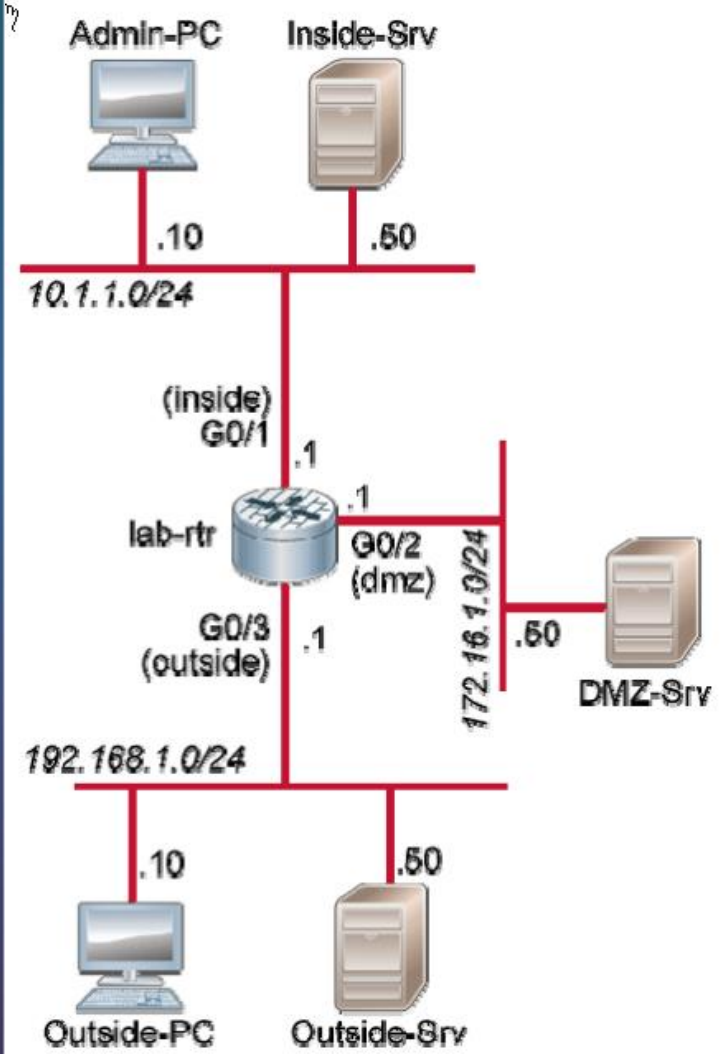
```
policy-map type inspect priv-pub-pmap
  class type inspect private-http
    inspect
  class type inspect private-ftp
    inspect
  class type inspect private-ssh
    inspect
  class type inspect private-netbios
    inspect
  class type inspect all-private
   inspect
  class class-default
!
zone security private
zone security public
zone-pair security priv-pub source private destination public
service-policy type inspect priv-pub-pmap
!
interface FastEthernet4
ip address 172.16.108.44 255.255.255.0
zone-member security public
!
interface Vlan1
ip address 192.168.108.1 255.255.255.0
zone-member security private
!
access-list 101 permit ip 192.168.108.0 0.0.0.255 any
```

Three zones example



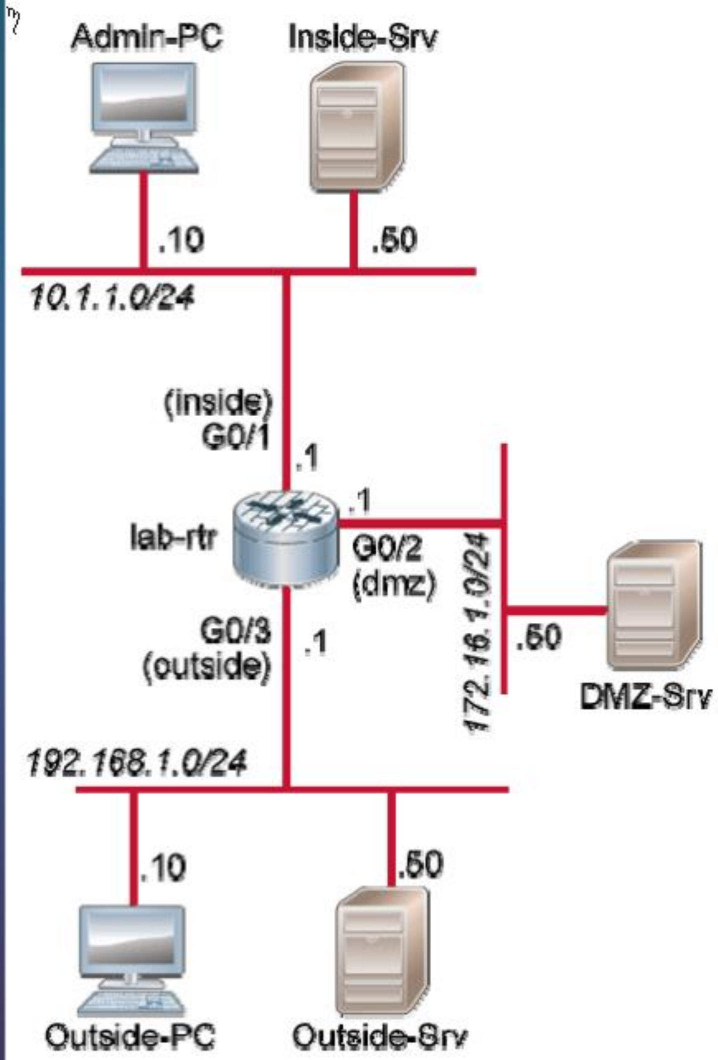

```
! – step 1 – define zones and assign int

lab-rtr>enable
lab-rtr#config t
Enter configuration commands, one per line. End with CNTL/Z.
lab-rtr(config)

lab-rtr(config)#zone security inside
lab-rtr(config-sec-zone)#description Trusted Internal Networks
lab-rtr(config-sec-zone)#exit
lab-rtr(config)#int gi0/1
lab-rtr(config-if)#zone-member security inside
lab-rtr(config-if)#exit

lab-rtr(config)#
lab-rtr(config)#zone security DMZ
lab-rtr(config-sec-zone)#description DMZ Services Available to Internet
lab-rtr(config-sec-zone)#exit
lab-rtr(config)#int gi0/2
lab-rtr(config-if)#zone-member security DMZ
lab-rtr(config-if)#exit

lab-rtr(config)#
lab-rtr(config)#zone security outside
lab-rtr(config-sec-zone)#description Untrusted Internet
lab-rtr(config-sec-zone)#exit
lab-rtr(config)#int gi0/3
lab-rtr(config-if)#zone-member security outside
lab-rtr(config-if)#end
lab-rtr#

! Veriy
lab-rtr#show zone security
```

Three zones example



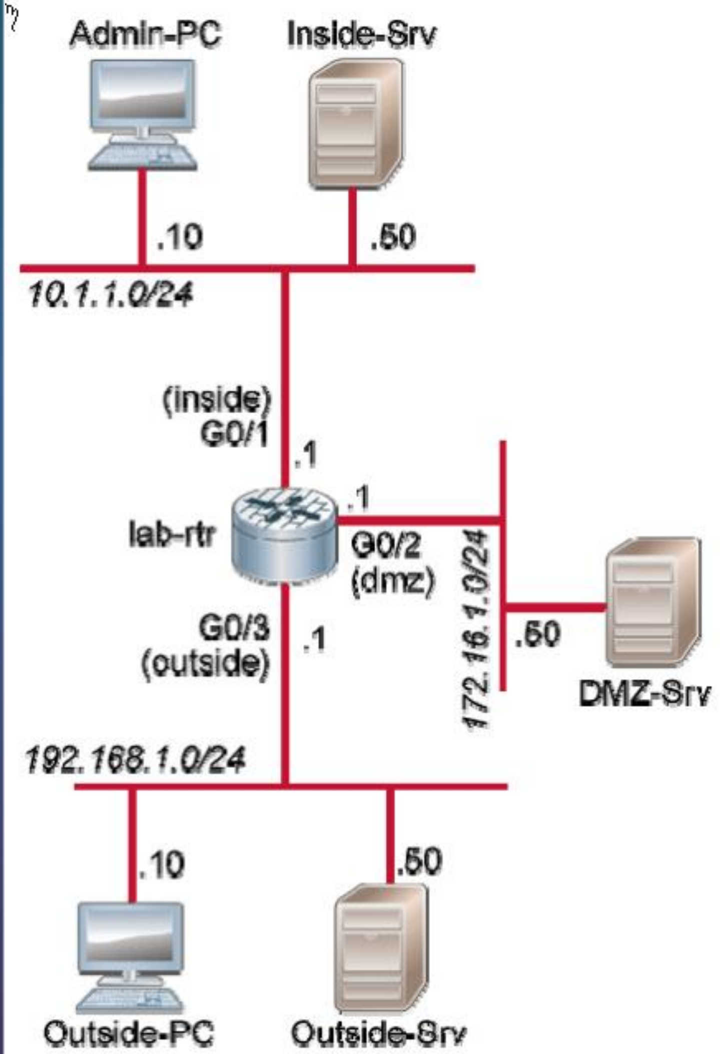

```
! - step 2 - Define and Verify Policies for DMZ Access
! Control access to DMZ

lab-rtr#config t
Enter configuration commands, one per line. End with CNTL/Z.
lab-rtr(config)#class-map type inspect match-any DMZ-Services
lab-rtr(config-cmap)#match protocol ftp
lab-rtr(config-cmap)#match protocol http
lab-rtr(config-cmap)#match protocol icmp
lab-rtr(config-cmap)#exit
lab-rtr(config)#

lab-rtr(config)#ip access-list extended DMZ-Servers
lab-rtr(config-ext-nacl)#permit ip any host 172.16.1.50
lab-rtr(config-ext-nacl)#exit
lab-rtr(config)#

lab-rtr(config)#class-map type inspect match-all Permitted-DMZ-Sessions
lab-rtr(config-cmap)#match class-map DMZ-Services
lab-rtr(config-cmap)#match access-group name DMZ-Servers
lab-rtr(config-cmap)#exit
lab-rtr(config)#

!policy map for DMZ
lab-rtr(config)#policy-map type inspect DMZ-Access-Policy
lab-rtr(config-pmap)#class type inspect Permitted-DMZ-Sessions
lab-rtr(config-pmap-c)#inspect
lab-rtr(config-pmap-c)#end
lab-rtr#


!verify
lab-rtr#show policy-map type inspect
```

Three zones example




```
! – step 2 – Define and Verify Policies for Inside - Outside access
! Control access from IN to OUT

lab-rtr#config t
Enter configuration commands, one per line. End with CNTL/Z.
lab-rtr(config)#class-map type inspect match-any Inside-To-Outside-
Protocols
lab-rtr(config-cmap)#match protocol ftp
lab-rtr(config-cmap)#match protocol tcp
lab-rtr(config-cmap)#match protocol icmp
lab-rtr(config-cmap)#exit
lab-rtr(config)#


lab-rtr(config)#policy-map type inspect Inside-To-Outside-Policy
lab-rtr(config-pmap)#class type inspect Inside-To-Outside-Protocols
lab-rtr(config-pmap-c)#inspect
lab-rtr(config-pmap-c)#exit
lab-rtr(config-pmap)#exit
lab-rtr(config)#


!verify
lab-rtr#show policy-map type inspect
```

Three zones example




```
! – step 3 – Define zone pairs

lab-rtr#config t
Enter configuration commands, one per line. End with CNTL/Z.
! Access to DMZ from outside
lab-rtr(config)#zone-pair security Outside-To-DMZ source outside
destination DMZ
lab-rtr(config-sec-zone-pair)#service-policy type inspect DMZ-Access-
Policy


! Access to DMZ from inside
lab-rtr(config)#zone-pair security Inside-To-DMZ source inside
destination DMZ
lab-rtr(config-sec-zone-pair)#service-policy type inspect DMZ-Access-
Policy


lab-rtr(config)#zone-pair security Inside-To-Outside source inside
destination outside
lab-rtr(config-sec-zone-pair)#service-policy type inspect Inside-To-
Outside-Policy
lab-rtr(config-sec-zone-pair)#end
lab-rtr#


!verify
lab-rtr#show zone-pair security
lab-rtr#show policy-map type inspect zone-pair Inside-To-DMZ sessions
```

ZPF Configuration Considerations

Source Interface Member of Zone?	Destination Interface Member of Zone?	Zone-Pair Exists?	Policy Exists?	Result
NO	NO	N/A	N/A	PASS
YES	NO	N/A	N/A	DROP
NO	YES	N/A	N/A	DROP
YES (private)	YES (private)	N/A	N/A	PASS
YES (private)	YES (public)	NO	N/A	DROP
YES (private)	YES (public)	YES	NO	PASS
YES (private)	YES (public)	YES	YES	INSPECT

- The default policy between zones is to deny all
- No filtering is applied for intra-zone traffic
- Only one zone is allowed per interface.
- An interface pair can be assigned only one policy.
- No Classic Firewall and ZPF configuration on the same interface is allowed.
- If only one zone member is assigned, all traffic is dropped.
- To permit traffic to and from a zone member interface, a policy allowing or inspecting traffic must be configured between that zone and any other zone.
- Only explicitly allowed traffic is forwarded between zones.
- Traffic to the self zone is not filtered.
- A zone must be configured before you can assign interfaces to the zone
- All active interfaces should be a member of a zone
- The rules are different when router is the source or destination of the traffic.

Resources

- Cisco IOS Firewall
  - https://www.cisco.com/c/en/us/support/security/ios-firewall/series.html
- Zone-Based Policy Firewall Design and Application Guide
  - https://www.cisco.com/c/en/us/support/docs/security/ios-firewall/98628-zone-design-guide.html?referring_site=RE&pos=1&page=https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_zbf/configuration/15-mt/sec-data-zbf-15-mt-book/sec-zone-pol-fw.html
- Security Configuration Guide: Zone-Based Policy Firewall, Cisco IOS Release 15M&T
  - https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_zbf/configuration/15-mt/sec-data-zbf-15-mt-book.html

UNIVERSITY OF ŽILINA
Faculty of Management Science and Informatics

KATEDRA INFORMAČNÝCH SIETÍ * FAKULTA RIADENIA A INFORMATIKY ŽU *

CISCO
Networking Academy

MINISTERSTVO
MINISTERSTVO
ŠKOLSTVA, VEDY, VÝSKUMU A ŠPORTU SLOVENSKEJ REPUBLIKY

Aktualizované v rámci projektu KEGA 026TUKE-4/2021.