CSPFA

Cisco Secure PIX Firewall Advanced

Student Guide

Version 1.01

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706 USA

cause interference with radio and television reception. This equipment has been tested and found to comply with the limits for a Class B digital device in accordance with the specifications in part 15 of the FCC rules. These specifications are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation.

Modifying the equipment without Cisco's written authorization may result in the equipment no longer complying with FCC requirements for Class A or Class B digital devices. In that event, your right to use the equipment may be limited by FCC regulations, and you may be required to correct any interference to radio or television communications at your own expense.

You can determine whether your equipment is causing interference by turning it off. If the interference stops, it was probably caused by the Cisco equipment or one of its peripheral devices. If the equipment causes interference to radio or television reception, try to correct the interference by using one or more of the following measures:

• Turn the television or radio antenna until the interference stops.

• Move the equipment to one side or the other of the television or radio.

• Move the equipment farther away from the television or radio.

• Plug the equipment into an outlet that is on a different circuit from the television or radio. (That is, make certain the equipment and the television or radio are on circuits controlled by different circuit breakers or fuses.)

Modifications to this product not authorized by Cisco Systems, Inc. could void the FCC approval and negate your authority to operate the product.

Cisco Secure PIX Firewall Advanced: Student Guide

1

Course Introduction

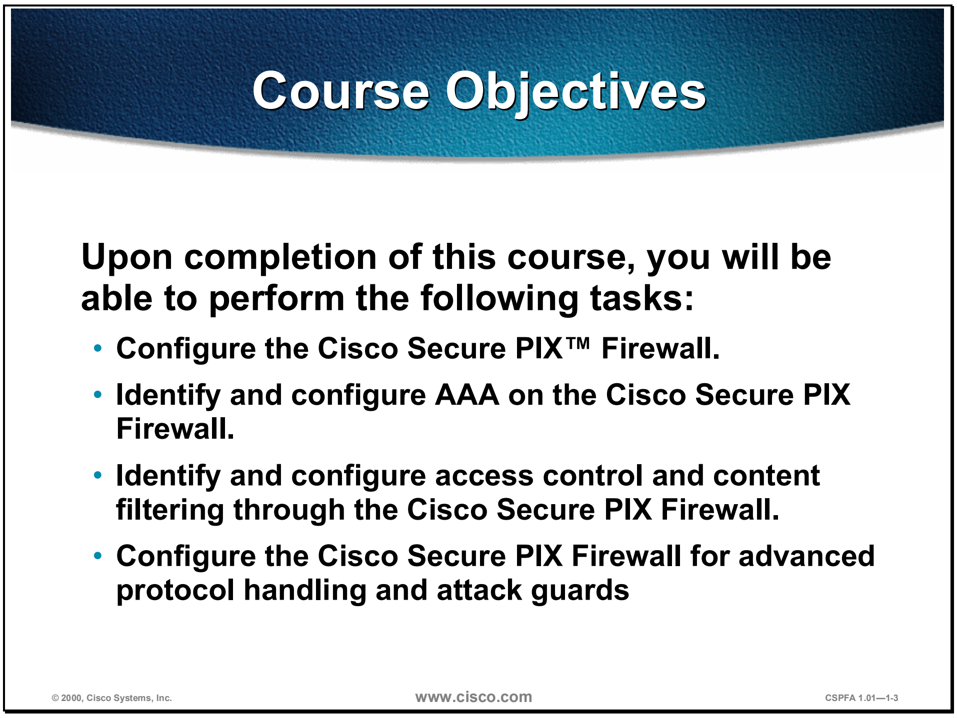
Overview

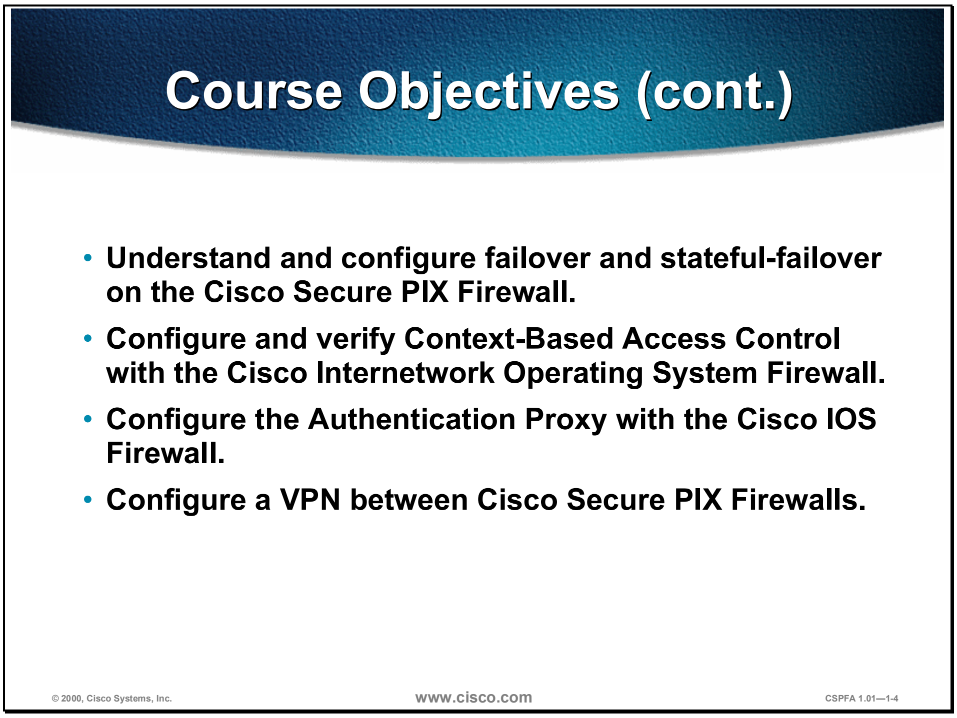
This chapter includes the following topics:

- Course agenda
- Lab topology overview
- Summary

Course Agenda

This section introduces the course and the course objectives.





Course Agenda

Day 1

- Chapter 1—Course Introduction
- Chapter 2—Cisco Secure PIX Firewall Configuration
- Chapter 3—Access Control Configuration and Content Filtering
- Lunch
- Chapter 4—AAA Configuration on the Cisco Secure PIX Firewall

Day 2

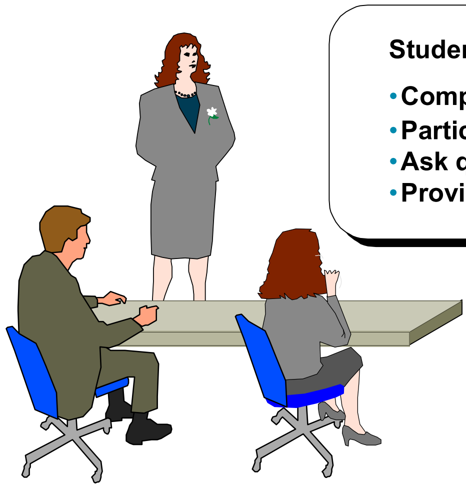
- Chapter 5—Cisco Secure PIX Firewall Advanced Protocol Handling and Attack Guards
- Chapter 6—Cisco Secure PIX Firewall Failover
- Lunch
- Chapter 7—Cisco Internetwork Operating System Firewall Context-Based Access Control Configuration
- Chapter 8—Cisco IOS Firewall Authentication Proxy Configuration

Course Agenda (cont.)

Day 3

- Chapter 8—Cisco IOS Firewall Authentication Proxy Configuration (cont.)
- Chapter 9—VPN Configuration with the Cisco Secure PIX Firewall
- Lunch
- Chapter 9—VPN Configuration with the Cisco Secure PIX Firewall (cont.)

Participant Responsibilities



Student Responsibilities

- Complete prerequisites
- Participate in lab exercises
- Ask questions
- Provide feedback

 www.cisco.com CSPFA 1.01—1-7

General Administration

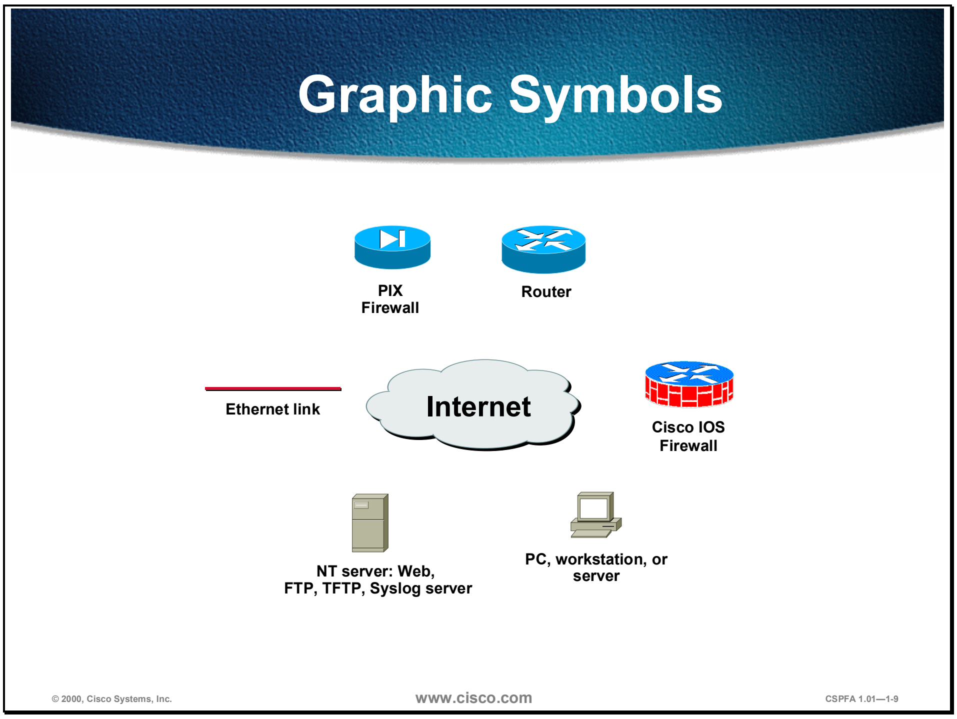
Class–related

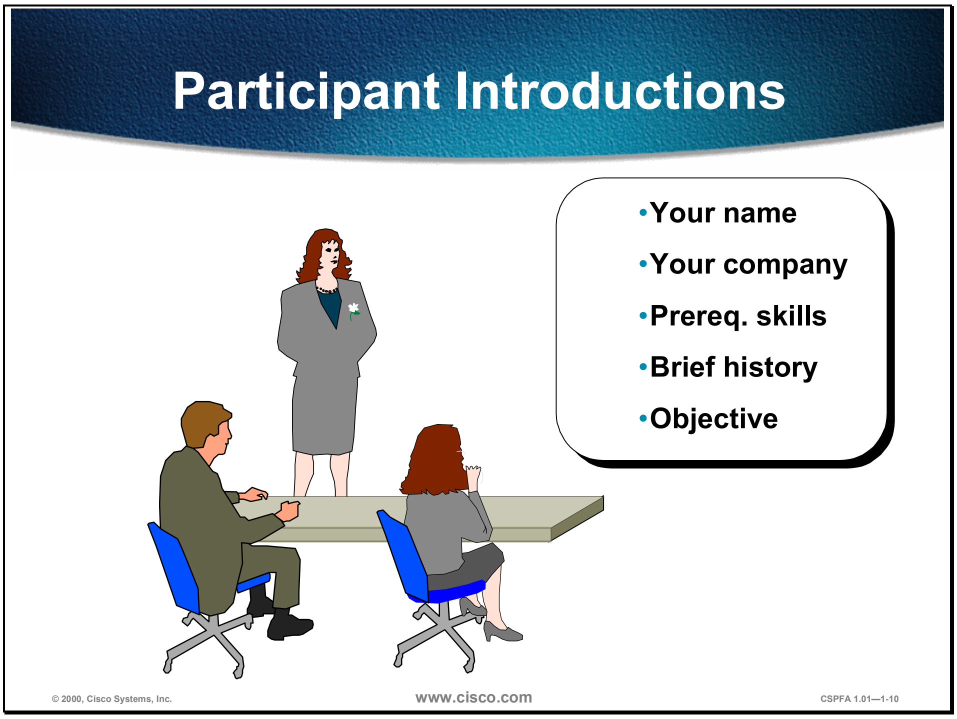
- Sign-in sheet
- Length and times
- Break and lunch room locations
- Attire

Facilities–related

- Participant materials
- Site emergency procedures
- Restrooms
- Telephones/faxes

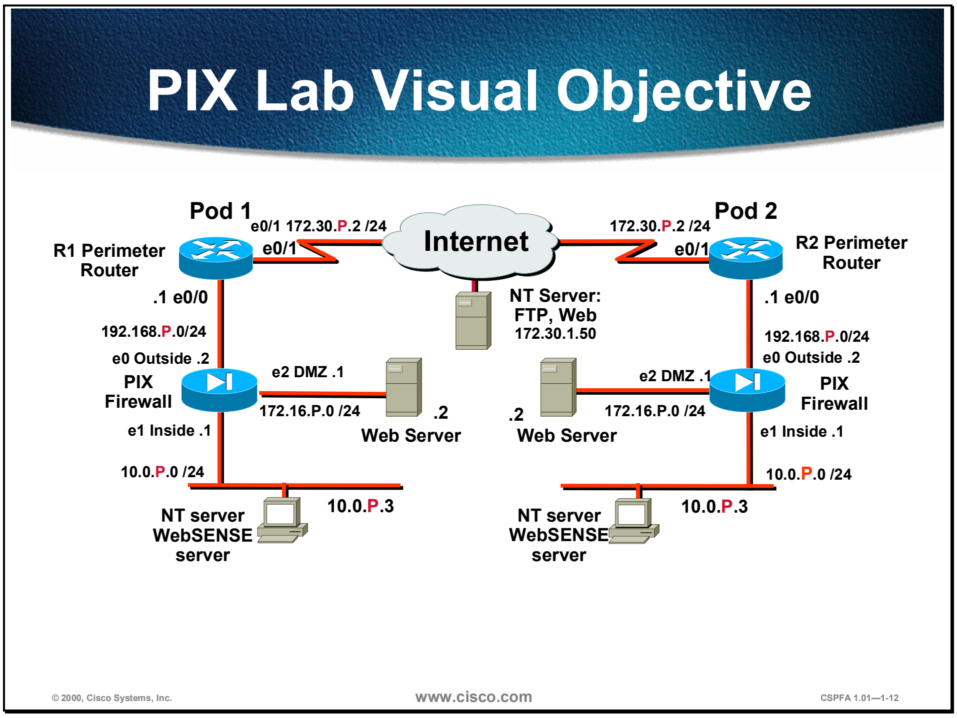
 www.cisco.com CSPFA 1.01—1-8

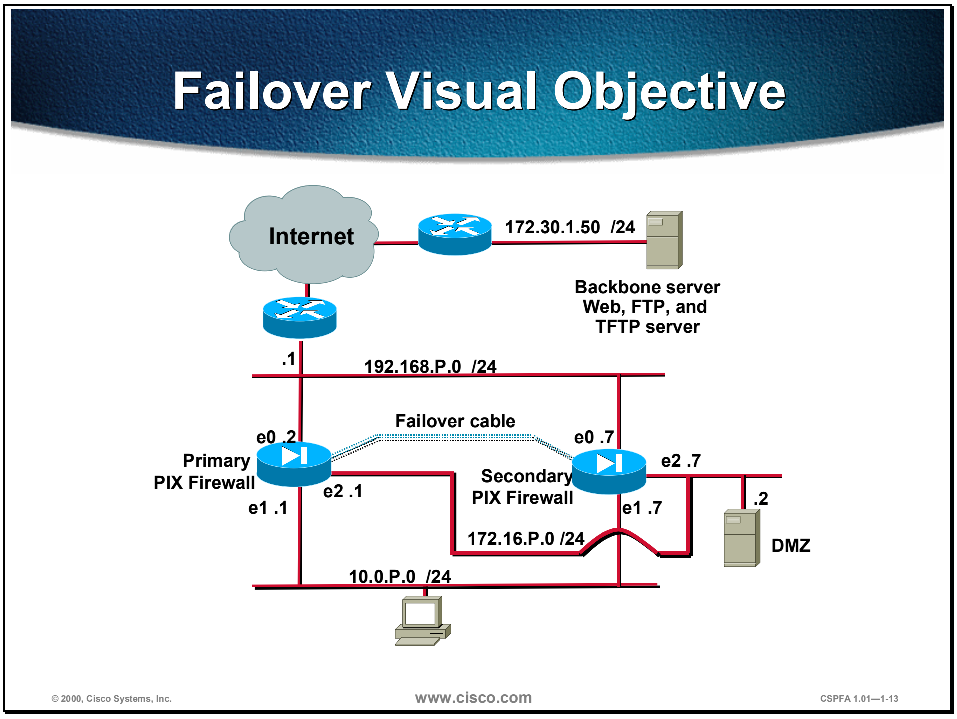

Graphic Symbols
PIX
Firewall
Router
Ethernet link
Internet
Cisco IOS
Firewall
NT server: Web,
FTP, TFTP, Syslog server
PC, workstation, or
server
© 2000, Cisco Systems, Inc.
www.cisco.com
CSPFA 1.01—1-9


Participant Introductions
•Your name
•Your company
•Prereq. skills
•Brief history
•Objective
© 2000, Cisco Systems, Inc.
www.cisco.com
CSPFA 1.01—1-10

Lab Topology Overview

This section explains the lab topology that is used in this course.





2

Cisco Secure PIX Firewall Configuration

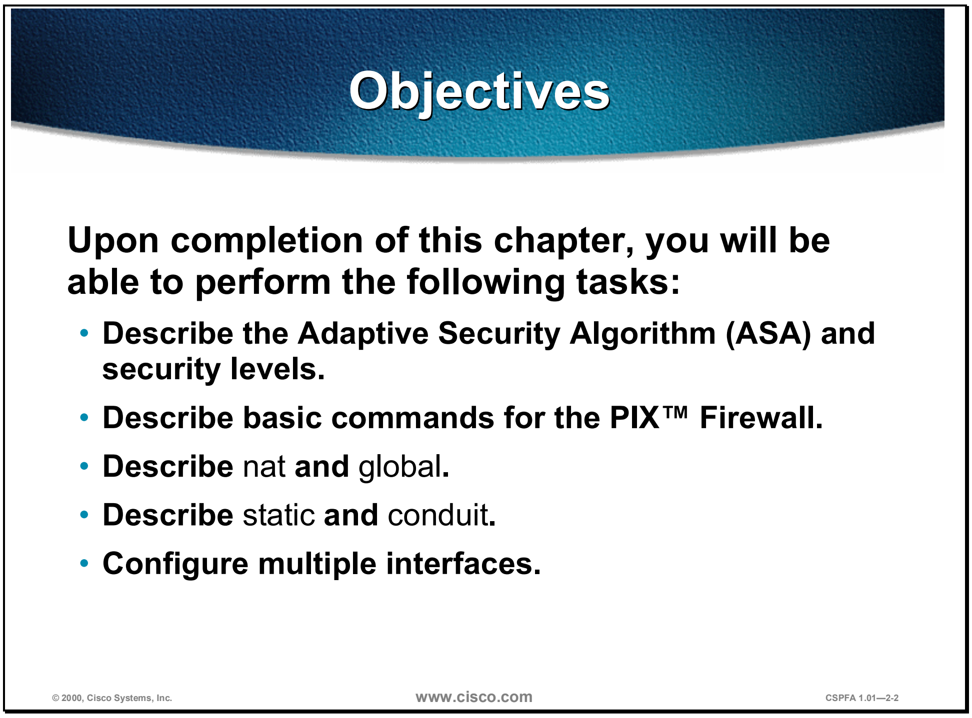
Overview

This chapter includes the following topics:

- Objectives
- Adaptive Security Algorithm
- Primary commands
- Access configuration through the PIX Firewall
- Multiple interface configuration
- Lab exercise
- Summary

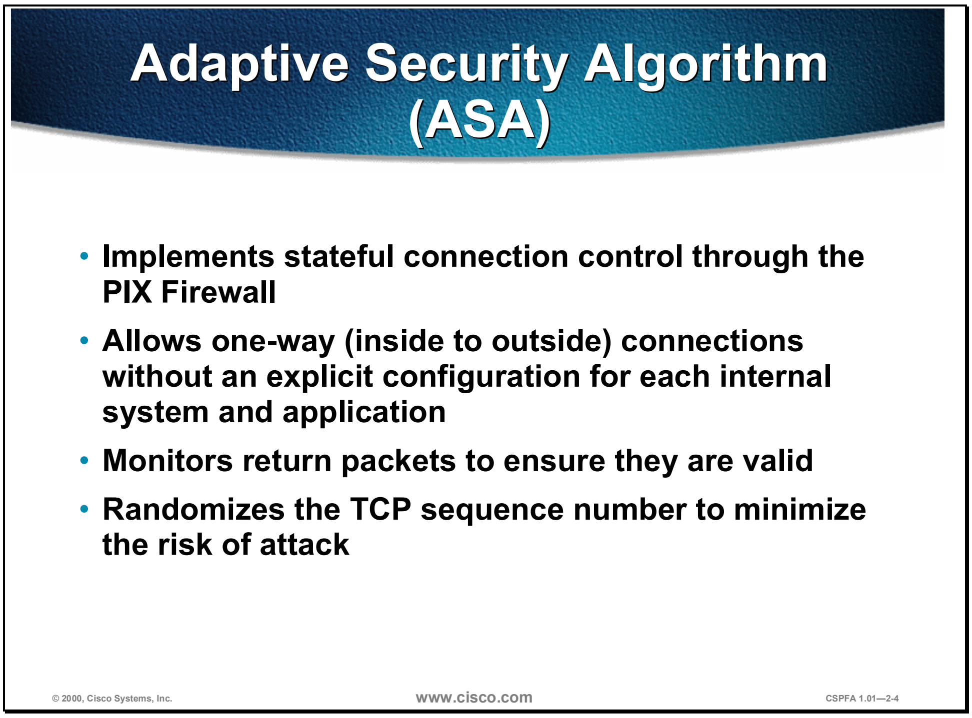
Objectives

This section lists the chapter's objectives.



Adaptive Security Algorithm

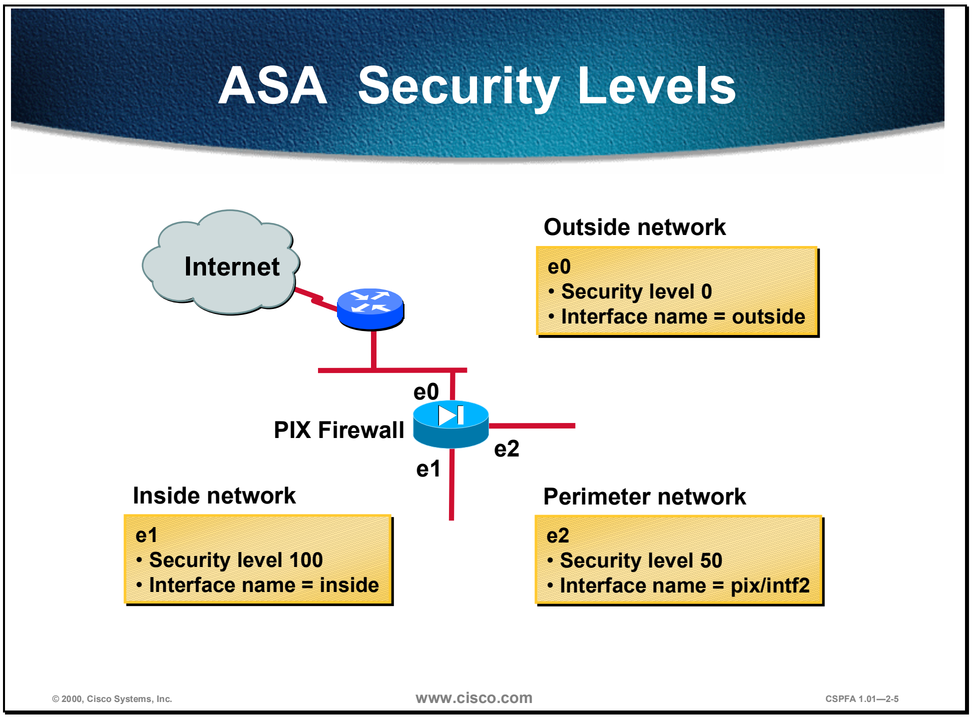
This section discusses the Adaptive Security Algorithm (ASA) and the ASA security levels.



The ASA is a stateful approach to security. Every inbound packet is checked against the ASA and against connection state information in the PIX™ Firewall's memory. Knowledge of the ASA is fundamental to implementing Internet access security because it performs the following tasks:

- Implements stateful connection control through the PIX Firewall
- Allows one-way (inside to outside) connections without an explicit configuration for each internal system application
- Monitors return packets to ensure they are valid
- Randomizes the TCP sequence number to minimize the risk of attack

ASA maintains the secure perimeters between the networks controlled by the PIX Firewall. The stateful connection-oriented ASA design creates session flows based on source and destination addresses. ASA randomizes TCP sequence numbers, port numbers, and TCP flags before the completion of the connection. This function is always running, monitoring return packets to ensure that they are valid.



The ASA security levels designate whether an interface is inside (trusted) or outside (untrusted) relative to another interface. An interface is considered inside in relation to another interface if its security level is higher than that of the other interface, and is considered outside in relation to another interface if its security level is lower than that of the other interface.

The primary rule for security levels is that an interface with a higher security level can access an interface with a lower security level. Conversely, an interface with a lower security level cannot access an interface with a higher security level without a conduit (discussed later). Security levels range from 0 to 100 The following are more specific rules for these security levels:

- Security level 100—This is the highest security level for the inside interface of the PIX Firewall. This is the default setting for the PIX Firewall and cannot be changed. Because 100 is the most trusted interface security level, your corporate network should be set up behind it. This is so no one else can access it unless they are specifically given permission, and every device behind this interface can have access outside of the corporate network.
- Security level 0—This is the lowest security level for the outside interface of the PIX Firewall. This is the default setting for the PIX Firewall and cannot be changed. Because 0 is the least trusted interface security level, you should set your most untrusted network behind this interface so that it does not have access to other interfaces unless it is specifically given permission. This interface is usually used for your Internet connection.
- Security levels 1–99—These are the security levels that you can assign to the perimeter interfaces connected to the PIX Firewall. You assign the security levels based on the type of access you want each device to have.

The following are examples of different interface connections between the PIX Firewall and other perimeter devices:

- More secure interface (the higher security level) to a less secure interface (the lower security level)—Traffic originating from the inside interface of the PIX Firewall with a security level of 100 to the outside interface of the PIX Firewall with a security level of 0 follows this rule: allow all IP-based traffic unless restricted by access lists, authentication, or authorization.
- Less secure interface (lower security level) to a more secure interface (higher security level)—Traffic originating from the outside interface of the PIX Firewall with a security level of 0 to the inside interface of the PIX Firewall with a security level of 100 follows this rule: drop all packets unless specifically allowed by the **conduit** command. Further restrict the traffic if authentication and authorization is used.
- Same secure interface to a same secure interface—No traffic flows between two interfaces with the same security level.

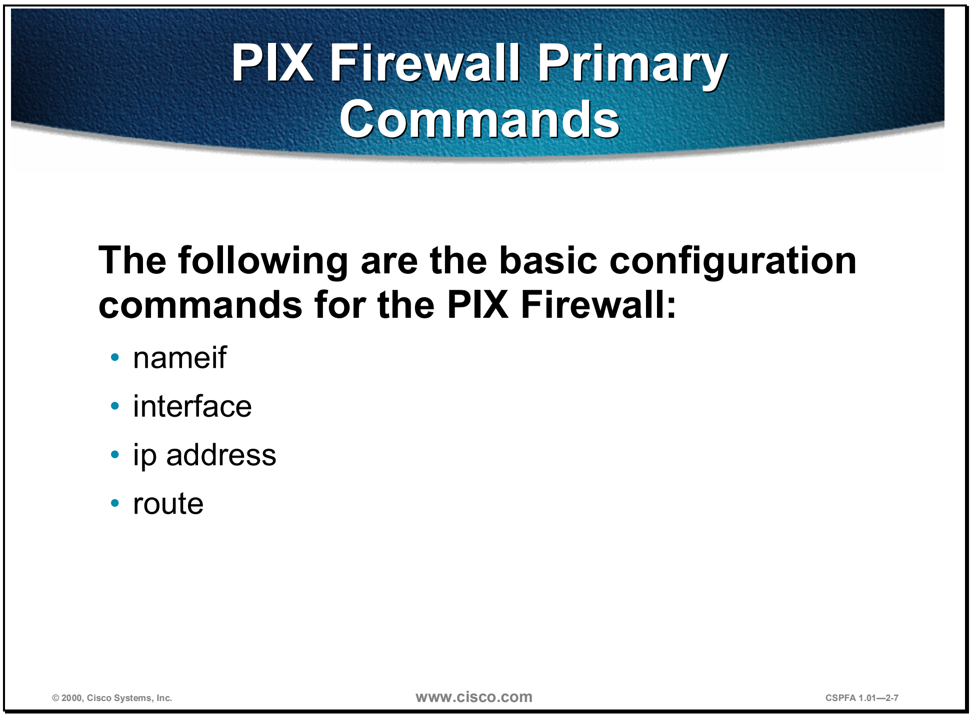
The following table explains the diagram in the previous figure.

Interface Pair	Relative Interface Relationship for Ethernet 2 (DMZ) Interface	Configuration Guidelines
Outside security 0 to DMZ security 50	DMZ is considered inside	Statics and conduits must be configured to enable sessions originated from the outside interface to the DMZ interface.
Inside security 100 to DMZ security 50	DMZ is considered outside	Globals and NAT are configured to enable sessions originated from the inside interface to the DMZ interface. Statics may be configured for the DMZ interface to ensure service hosts have same source address.

Note The PIX Firewall can have up to four perimeter networks for a total of six interfaces.

Primary Commands

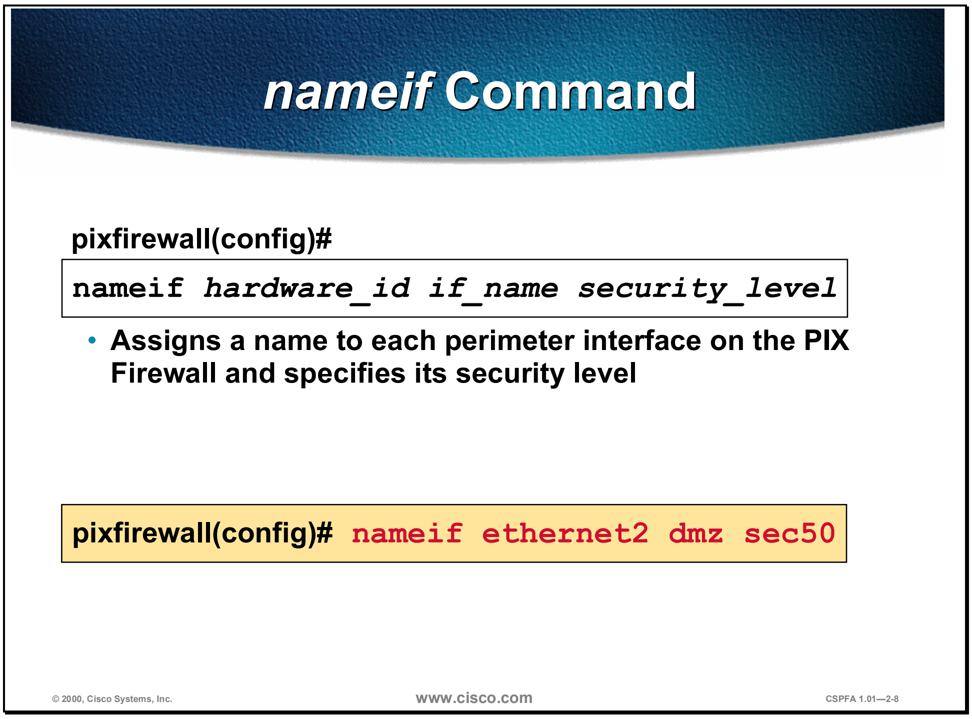
This section discusses the basic configurations necessary to run the PIX Firewall.



The basic PIX Firewall commands are as follows:

- **nameif**—Assigns a name to each interface and specifies a security level for each interface.
- **interface**—Configures the type and capability of each perimeter interface.
- **ip address**—Assigns an IP address to each interface.
- **route**—Defines a static or default route for an interface.

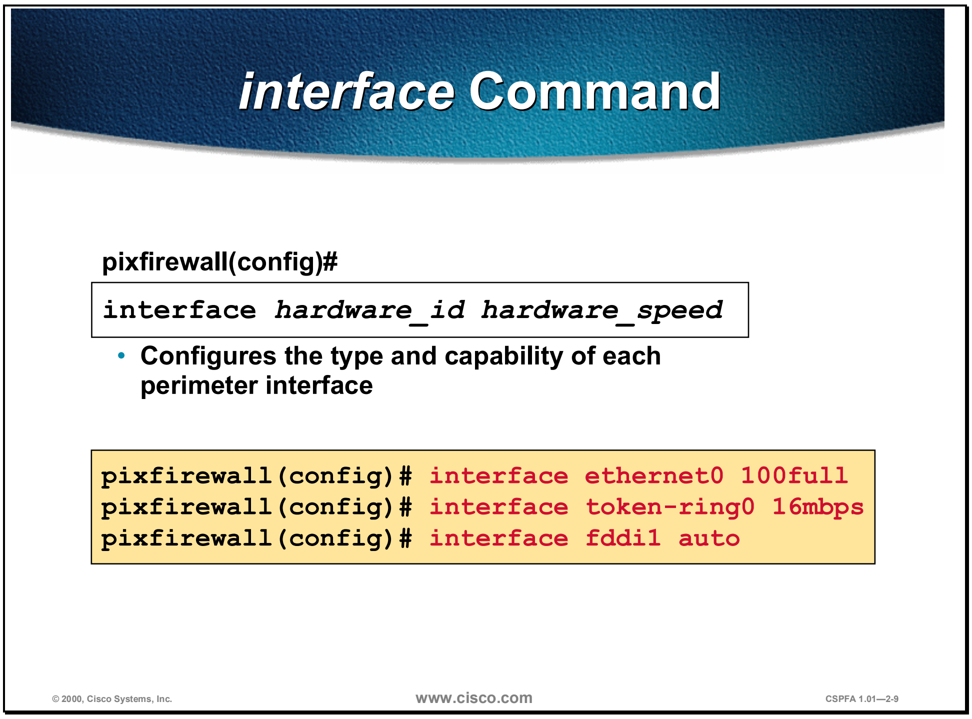
Note Inside and outside interface names can be changed, but this is not recommended. These terms are used so universally, that it would be confusing to others.



The **nameif** command assigns a name to each perimeter interface on the PIX Firewall and specifies its security level (except for the inside and outside PIX Firewall interfaces, which are named by default). The syntax for the **nameif** command is as follows:

nameif *hardware_id if_name security_level*

Argument	Description
hardware_id	Specifies a perimeter interface and its slot location on the PIX Firewall. You can enter three interfaces here: Ethernet, FDDI, or Token Ring. Each interface is represented by an alphanumeric identifier based on which interface it is and what numeric identifier you choose to give it. For example, an Ethernet interface is represented as e1, e2, e3, and so on; a FDDI interface is represented as fddi1, fddi2, fddi3, and so on; and a Token Ring interface is represented as token-ring1, token-ring2, and token-ring3, and so on.
if_name	Describes the perimeter interface. This name is assigned by you, and must be used in all future configuration references to the perimeter interface.
security_level	Indicates the security level for the perimeter interface. Enter a security level of 1–99.



The **interface** command identifies interfaces, sets the interface speed, and enables the interface to function.

The syntax for the **interface** command is as follows:

interface *hardware_id hardware_speed*

Argument	Description
hardware_id	Specifies an interface and its slot location on the PIX Firewall. This is the same variable used with the **nameif** command.
hardware_speed	Determines the connection speed. Enter **`auto`** so the PIX Firewall will sense the speed needed for the device.

Note When a FDDI or Token Ring interface card is installed using the **interface** command, you must define the FDDI or Token Ring interface card because the PIX Firewall does not automatically recognize them.

ip address Command

pixfirewall(config)#

```
ip address if_name ip_address [netmask]
```

- **Assigns an IP address to each interface**

```
pixfirewall(config)# ip address dmz
 172.16.2.1 255.255.255.0
```

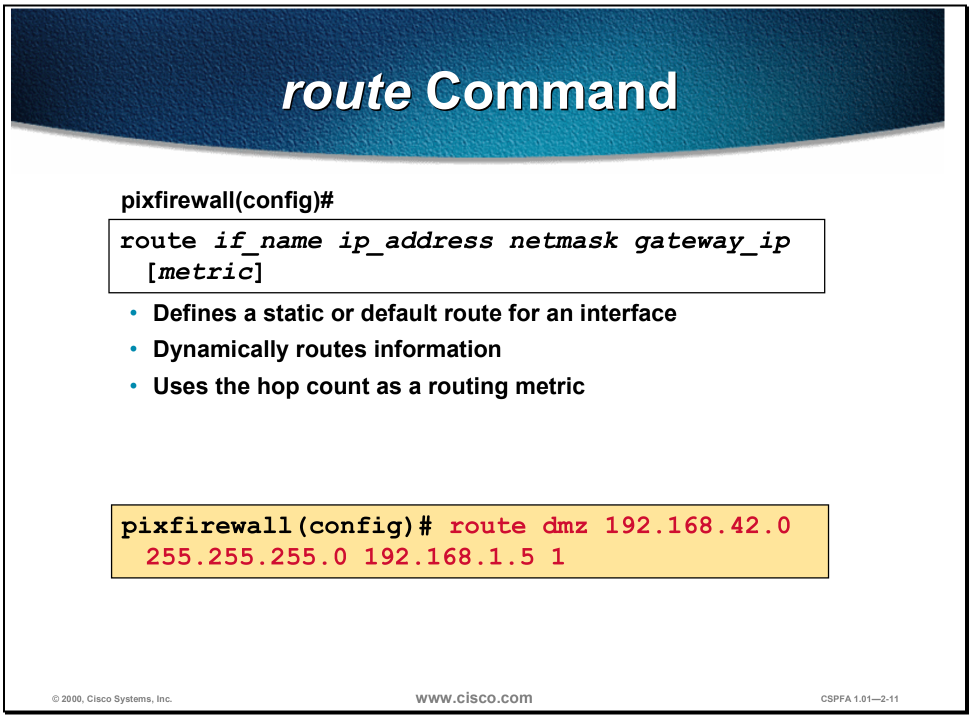
© 2000, Cisco Systems, Inc. www.cisco.com CSPFA 1.01—2-10

Each interface on the PIX Firewall must be configured with an IP address. The syntax for the **ip address** command is as follows:

ip address ***if_name ip_address*** **[*netmask*]**

Argument	Description
if_name	Describes the interface. This name is assigned by you, and must be used in all future configuration references to the interface.
ip_address	The IP address of the interface.
netmask	If a network mask is not specified, the default network mask is assumed.

After configuring the IP address and netmask, use the **show ip** command to view which addresses are assigned to the network interfaces. If you made a mistake while entering the information, reenter the command with the correct information.



The syntax for the **route** command is as follows:

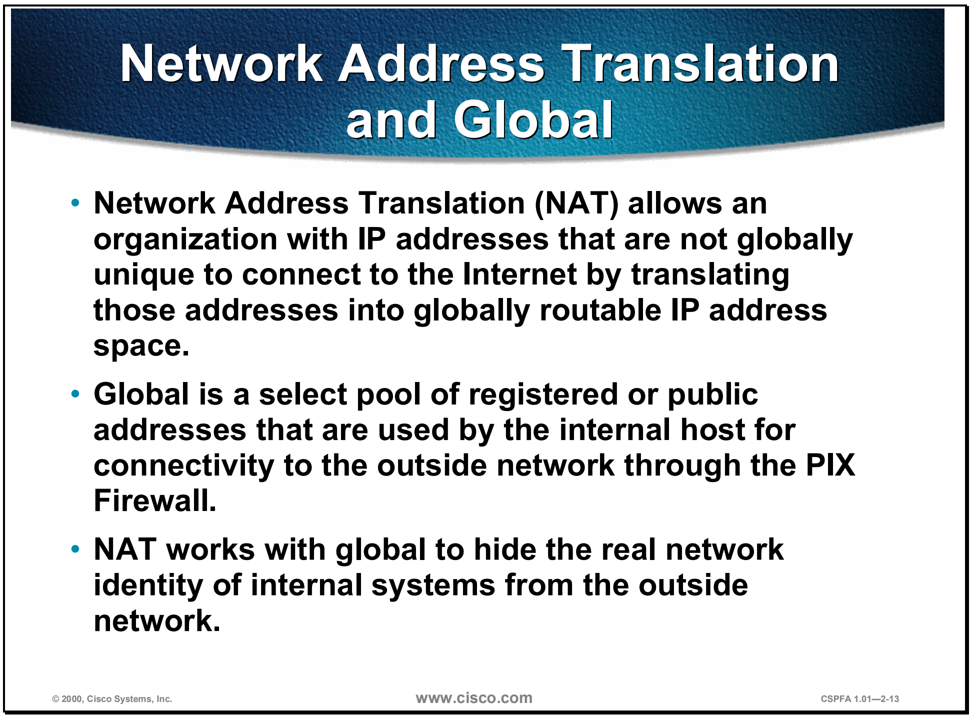
route ***if_name ip_address netmask gateway_ip*** **[*metric*]**

Argument	Description
if_name	Describes the internal or external network interface name.
ip_address	Describes the internal or external network IP address. Use *0.0.0.0* to specify a default route. The *0.0.0.0* IP address can be abbreviated as *0*.
netmask	Specifies a network mask to apply to ***ip_address***. Use *0.0.0.0* to specify a default route. The *0.0.0.0* netmask can be abbreviated as *0*.
gateway_ip	Specifies the IP address of the gateway router (the next hop address for this route).
metric	Specifies the number of hops to ***gateway_ip***. If you are not sure, enter **1**. Your WAN administrator can supply this information or you can use a **traceroute** command to obtain the number of hops. The default is *1* if a metric is not specified.

All routes entered using the **route** command are stored in the configuration when it is saved. In the example shown in the figure, all outgoing packets are sent to the 192.168.1.1 router IP address. More than one route can be configured.

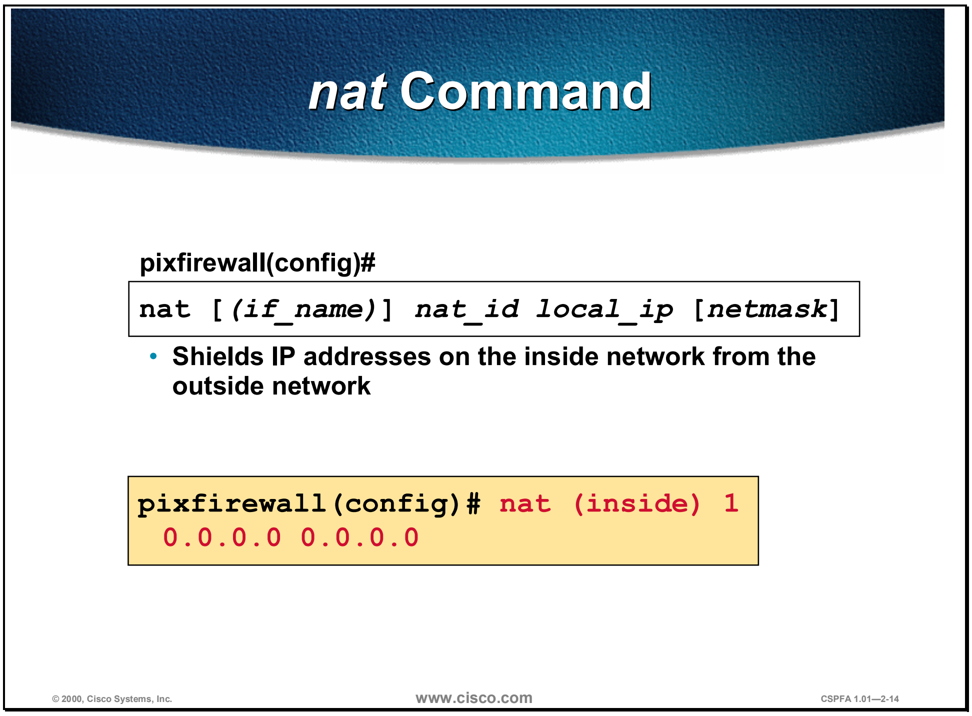
Access Configuration Through the PIX Firewall

This section discusses ways to configure the PIX Firewall to allow access through the PIX Firewall.



The **nat** command lets you enable or disable address translation for one or more internal addresses. Address translation means that when a host starts an outbound connection, the IP addresses in the internal network are translated into global addresses. Address translation lets your network have any IP addressing scheme. The PIX Firewall protects these addresses from visibility on the external network. You can have up to 1,000 Network Address Translation (NAT) groups.

- NAT allows an organization with IP addresses that are not globally unique to connect to the Internet by translating those addresses into globally routable IP address space.
- Global is a select pool of registered or public addresses that are used by the internal host for connectivity to the outside network through the PIX Firewall.
- The **nat** command works with the **global** command to hide the real network identity of internal systems from the outside network.



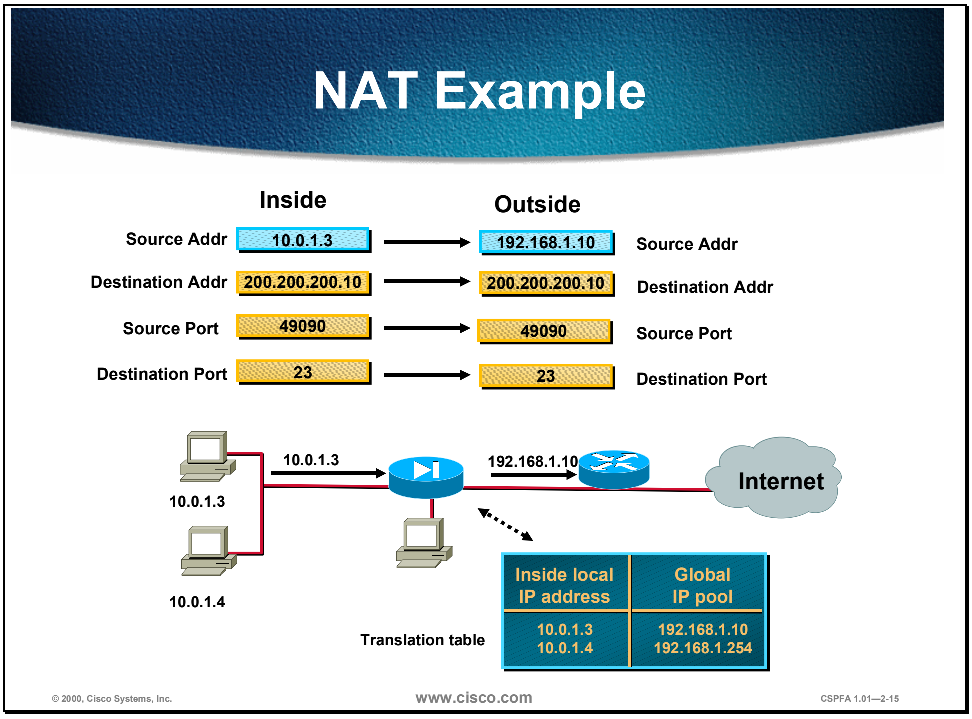
NAT enables you to keep your internal IP addresses—those behind the PIX Firewall—unknown to external networks. NAT accomplishes this by translating the internal IP addresses, which are not globally unique, into globally accepted IP addresses before packets are forwarded to the external network.

The syntax for the **nat** command is as follows:

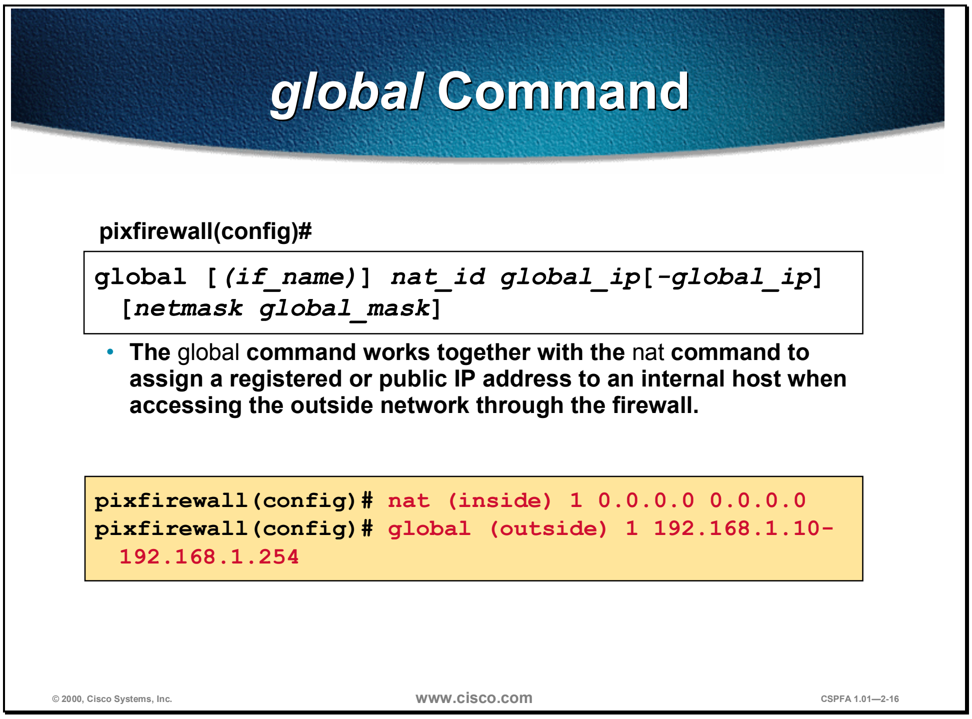
nat [*(if_name)*] *nat_id local_ip* [*netmask*]

Argument	Description
if_name	Describes the external network interface name where you will use the global addresses.
nat_id	Identifies the global pool and matches it with its respective **nat** command.
local_ip	The IP address that is assigned to the interface on the inside network.
netmask	Network mask for the local IP address. You can use 0.0.0.0 to allow all outbound connections to translate with IP addresses from the global pool.

When you initially configure the PIX Firewall, you can enable all inside hosts to access outbound connections with the **nat 1 0.0.0.0 0.0.0.0** command. The **nat 1 0.0.0.0 0.0.0.0** command enables NAT and lets all inside hosts (specified as *0.0.0.0*) to access outbound connections. The **nat** command can specify single hosts or ranges of hosts to make access more selective. You can use *0* in place of *0.0.0.0*.



When an outbound IP packet that is sent from a device on the inside network reaches the PIX Firewall, the source address is extracted and compared to an internal table of existing translations. If the device's address is not already in the table, it is translated and a new entry is created for that device and it is assigned a global IP address from a pool of global IP addresses. The table is then updated and the translated IP packet is forwarded. After the session terminates or there have been no translated packets for that particular IP address, the entry is removed from the table, and the global address is freed for use by another inside device.



The **global** command defines a pool of global addresses. The global addresses in the pool provide an IP address for each outbound connection and for those inbound connections resulting from outbound connections. Ensure that associated **nat** and **global** command statements have the same ***nat_id***.

The syntax for the **global** command is as follows:

global [*(if_name)*] *nat_id global_ip* [-*global_ip*] [*netmask global_mask*]

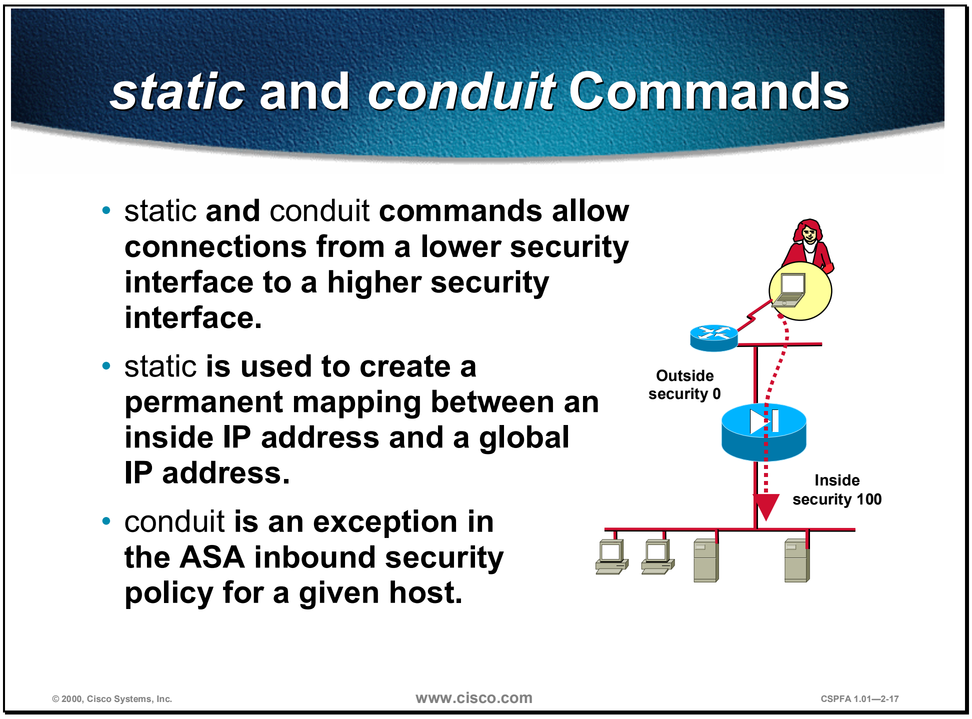
Argument	Description
if_name	Describes the external network interface name where you will use the global addresses.
nat_id	Identifies the global pool and matches it with its respective **nat** command.
global_ip	Single IP addresses or the beginning IP address for a range of global IP addresses.
global_ip	A range of global IP addresses.
netmask global_mask	The network mask for the global IP. If subnetting is in effect, use the subnet mask; for example, 255.255.255.128. If you specify an address range that overlaps subnets with the **netmask** command, this command will not use the broadcast or network address in the pool of global addresses. For example, if you use 255.255.255.128 and an address range of 192.150.50.20–192.150.50.140, the 192.150.50.127 broadcast address and the 192.150.50.128 network address will not be included in the pool of global addresses.

If the **nat** command is used, the companion command, **global**, must be configured to define the pool of translated IP addresses.

To delete a global entry, use the **no global** command. For example, **`no global (outside) 1 192.168.1.10-192.168.1.254 netmask 255.255.0.0`**

Note The PIX Firewall assigns addresses from the global pool starting from the low end to the high end of the range specified in the **global** command.

Note The PIX Firewall uses the global addresses to assign a virtual IP address to an internal NAT address. After adding, changing, or removing a global statement, use the **clear xlate** command to make the IP addresses available in the translation table.

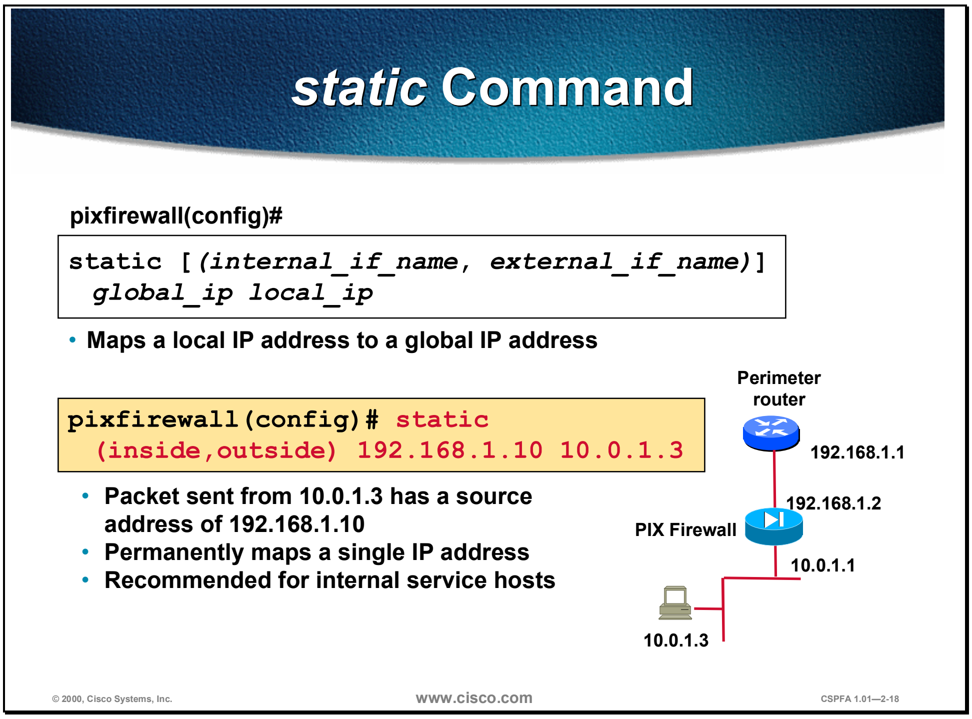


Although most connections occur from an interface with a high security level to an interface with a low security level, there are times when you will want to allow connections from an interface with a lower security level to an interface with a higher security level. To do this, use the **static** and **conduit** commands.

The **static** command creates static mapping between an inside IP address and a global IP address. Using the **static** command enables you to set a permanent global IP address for a particular inside IP address. This creates an entrance for the specified interfaces with the lower security level into the specified interface with a higher security level.

After creating a static mapping between an inside IP address and a global IP address by using the **static** command, the connection from the outside interface to the inside interface is still blocked by the PIX Firewall's ASA. The **conduit** command allows traffic to flow between interfaces and creates the exceptions to the PIX Firewall's ASA.

Note When you use a **static** command, you must also use a **conduit** command. The **static** command makes the mapping, and the **conduit** command lets users access the static mapping.



The **static** command creates a permanent mapping (called a static translation slot or "xlate") between a local IP address and a global IP address. For outbound connections, use **static** to specify an address in the pool of global addresses that is always used for translation between the local host and the global address. For inbound connections, use **static** with the **conduit** command to identify addresses visible on the external network.

The **static** command creates a permanent mapping (static translation slot) between a local IP address and a NIC-registered IP address. The syntax for the static command is as follows:

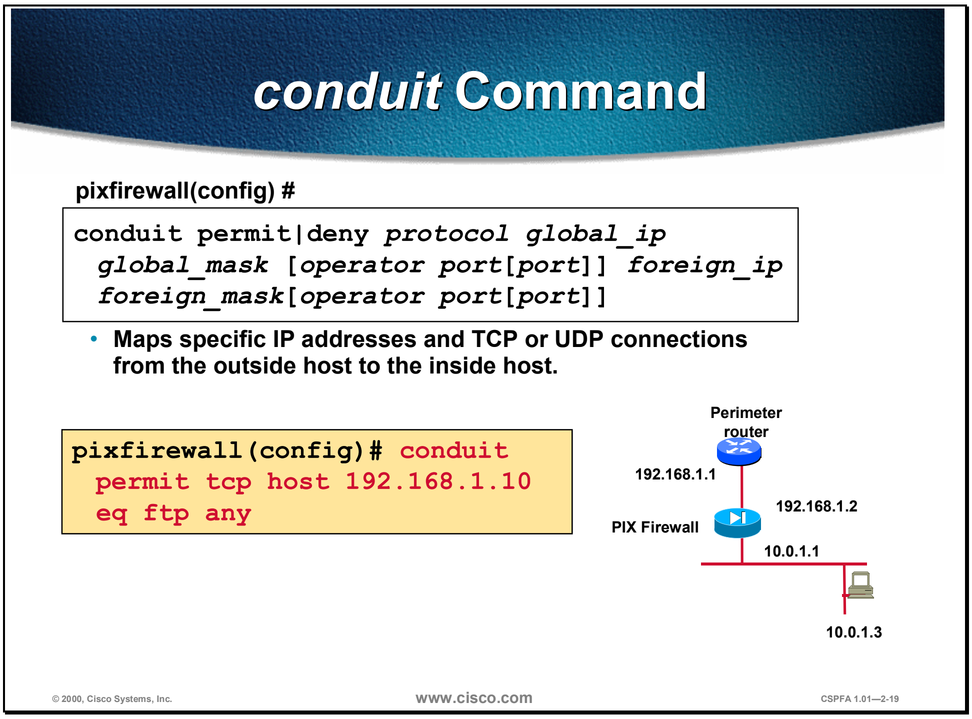
static [*(internal_if_name, external_if_name)*] *global_ip local_ip*

Argument	Description
internal_if_name	The internal network interface name.
external_if_name	The external network interface name.
global_ip	A global IP address for the outside interface. This address cannot be a PAT IP address.
local_ip	The local IP address from the inside network.
netmask	Reserve word required before specifying the network mask.

The security level for each interface is set by the **nameif** command. The **static** command allows traffic to originate from an interface with a lower security value, through the PIX Firewall, to an interface with a higher security value. For example, a static and conduit must be configured to allow incoming sessions from the outside interface to the DMZ interface, or from the outside interface to the inside interface.

Statics take precedence over **nat** and **global** command pairs. Use the **show static** command to view static statements in the configuration.

In the previous figure, when a packet from the client station 10.0.1.3 goes out through the PIX Firewall, it will have the source IP address of 192.168.1.10.



The **conduit** command permits or denies connections from outside the PIX Firewall to access TCP or UDP services on hosts inside the network. The conduit statement creates an exception to the PIX Firewall ASA by permitting connections from one PIX Firewall network interface to access hosts on another.

To allow connections from a lower security interface to a higher security interface, you must use the **conduit** command. The **conduit** command is what actually creates an exception to the standard PIX Firewall ASA.

The syntax for the **conduit** command is as follows:

conduit permit | **deny** ***protocol global_ip global_mask*** **[*operator port* [*port*]] *foreign_ip foreign_mask*** **[*operator port* [*port*]]**

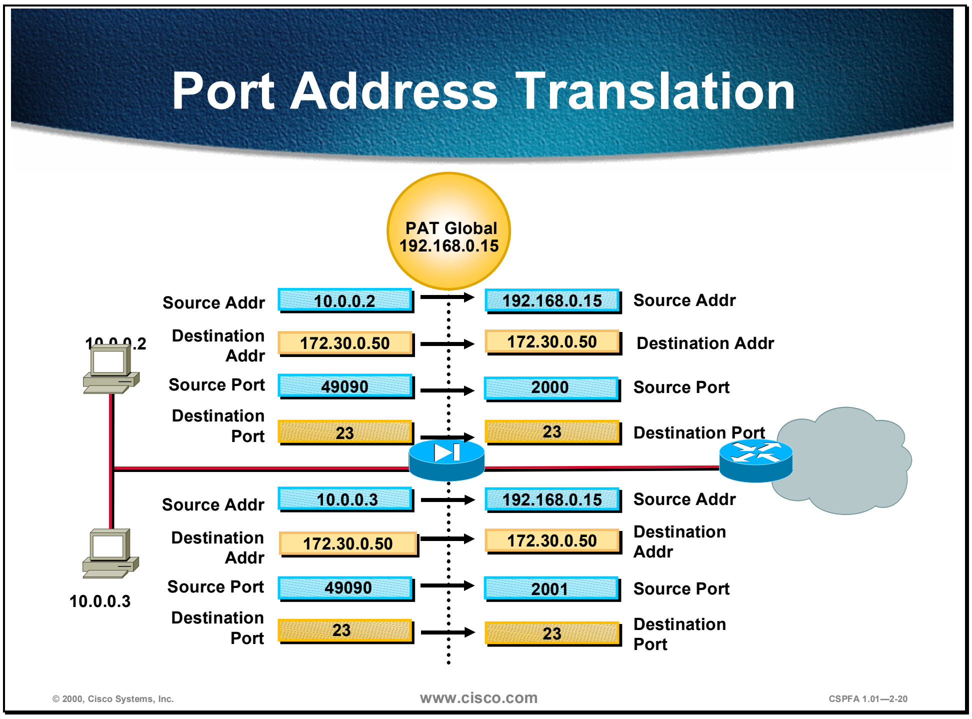
Argument	Description
permit	Permits access if the conditions are met.
deny	Denies access if the conditions are met.
protocol	Specifies the transport protocol for the connection. Possible literal values are **eigrp**, **gre**, **icmp**, **igmp**, **grp**, **ip**, **ipinip**, **nos**, **ospf**, **tcp**, **udp**, or an integer in the range 0 to 255 representing an IP protocol number. Use **ip** to specify all transport protocols. You can view valid protocol numbers online at: http://www.isi.edu/in-notes/iana/assignments/protocol-numbers
icmp	Permits or denies ICMP access to one or more global IP addresses. Specify the ICMP type in the ***icmp_type*** variable, or omit it to specify all ICMP types.
global_ip	A global IP address previously defined by a global or static command. You can use any if the ***global_ip*** and ***global_mask*** are *0.0.0.0 0.0.0.0*. The **any** option applies the permit or deny to the global addresses on all interfaces. If ***global_ip*** is a host, you can omit ***global_mask*** by specifying the **host** command before ***global_ip***.

Argument	Description
operator	A comparison operand that lets you specify a port or a port range. Possible values are **eq**, **t**, **any**, **gt**, **neq**, and **range**. Use the **no** operator and **port** to indicate all ports.
global_mask	Network mask of ***global_ip***. The ***global_mask*** is a 32-bit, 4-part dotted decimal; such as, 255.255.255.255. Use zeros to indicate bit positions to be ignored. Use subnetting if required. If you use 0 for ***global_ip***, use 0 for the ***global_mask***; otherwise, enter the ***global_mask*** appropriate to ***global_ip***.
port	Services you permit to be used while accessing ***global_ip***. Specify services by the port that handles it, such as 25 for SMTP, 80 for HTTP, and so on. 0 means any port. The port values are defined in RFC 1700. Permitted literal names are **dns**, **esp**, **ftp**, **h323**, **http**, **ident**, **nntp**, **ntp**, **pop2**, **pop3**, **pptp**, **rpc**, **smtp**, **snmp**, **snmptrap**, **sqlnet**, **tcp**, **telnet**, **tftp**, and **udp**. Note that you can specify literals in port ranges, for example, ftp-h323. You can also specify numbers.
foreign_ip	An external IP address (host or network) that can access the ***global_ip***. You can specify *0.0.0.0* or *0* for any host. If both the ***foreign_ip*** and ***foreign_mask*** are *0.0.0.0 0.0.0.0*, you can use the shorthand **any** command, which applies to all interfaces. If ***foreign_ip*** is a host, you can omit ***foreign_mask*** by specifying the **host** command before ***foreign_ip***.
foreign_mask	Network mask of ***foreign_ip***. The ***foreign_mask*** is a 32-bit, 4-part dotted decimal, such as 255.255.255.255. Use zeros in a part to indicate bit positions to be ignored. Use subnetting if required. If you use *0* for ***foreign_ip***, use *0* for the ***foreign_mask***; otherwise, enter the ***foreign_mask*** appropriate to ***foreign_ip***.
operator	A comparison operand that lets you specify a port or a port range. Possible values are **eq**, **lt**, **any**, **gt**, **neq**, **range**. Use the **no** operator and **port** to indicate all ports.
port	Services you permit to be used while accessing ***global_ip*** or ***foreign_ip***. Specify services by the port that handles it, such as 25 for SMTP, 80 for HTTP, and so on. You can specify ports by either a literal name or as a number in the range of 0 to 65535. You can specify all ports by not specifying a port value (for example, **conduit deny tcp any any**). This command is the default condition for the **conduit** command in that all ports are denied until explicitly permitted. You can view valid port numbers online at: http://www.isi.edu/in-notes/iana/assignments/port-numbers

The example in the previous figure allows FTP services via IP address 192.168.1.10 to the inside host 10.0.1.3 from the outside. The ***global_ip*** and ***global_mask*** arguments define the IP address or addresses where connections are being permitted.

You can have up to 8000 conduits, and can remove a conduit with the **no conduit** command.

Note If you want internal users to be able to ping external hosts, you must create an Internet Control Message Protocol (ICMP) conduit for echo reply; for example, to give ping access to all hosts, use the **conduit permit icmp any any** command. However, this may cause a lot of traffic on busy networks.

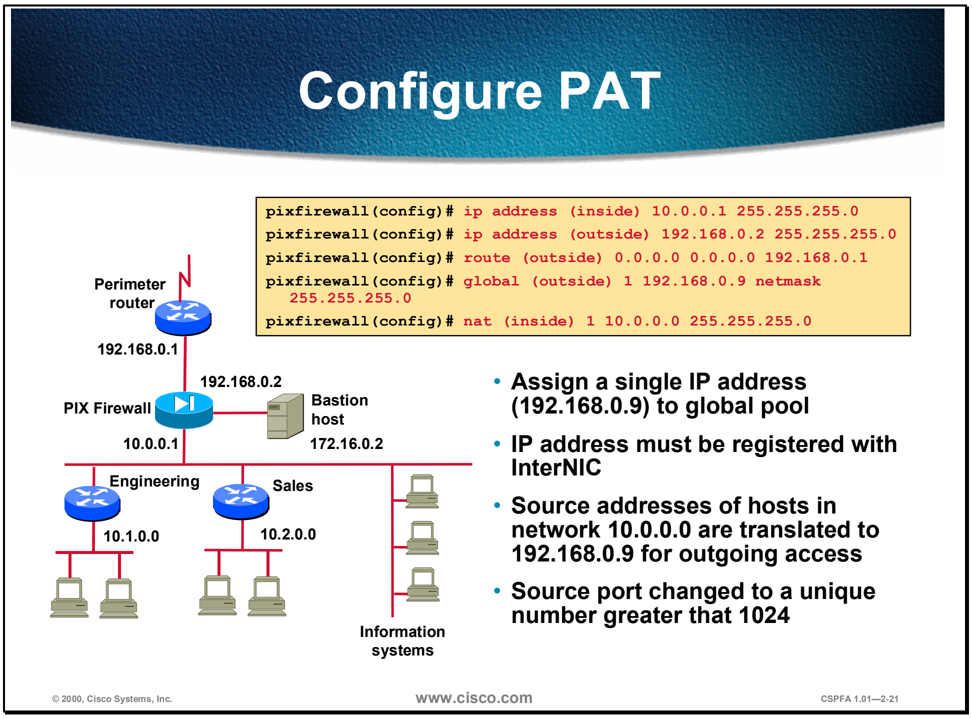


Port Address Translation (PAT) is a combination of an IP address and a source port number, which creates a unique session. PAT uses the same IP address for all packets, but a different unique source port greater than 1024.

PAT provides the following advantages:

- PAT and NAT can be used together.
- The PAT address is different than the outside interface address.
- PAT provides for IP address expansion.
- One outside IP address is used for up to 63,000 inside hosts.
- PAT maps port numbers to a single IP address.
- PAT hides the inside source address by using a single IP address from the PIX Firewall.

In the figure above, two clients are requesting connectivity to the Internet. The PIX Firewall checks security rules to verify the security levels, and then replaces the source IP address with the PAT IP address. To maintain accountability, the source port address is changed to a unique number greater than 1024.



The PIX Firewall PAT feature expands an address pool:

- One outside IP address is used for approximately 4,000 inside hosts (the practical limit is 4,000, and the theoretical limit is greater than 64,000)
- Maps TCP port numbers to a single IP address
- Hides the inside source address by using single IP address from the PIX Firewall
- PAT can be used with NAT
- A PAT address is a virtual address, different from the outside address

---

Note Do not use PAT when running multimedia applications through the PIX Firewall. Multimedia applications need access to specific ports and can conflict with port mappings provided by PAT.

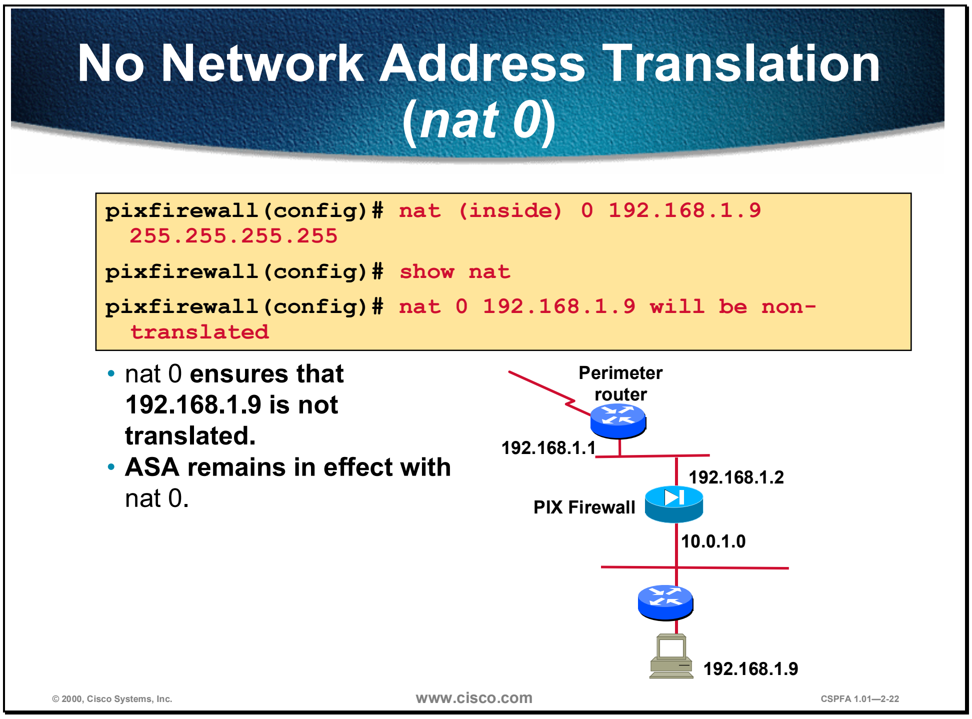
In the example of PAT in the preceding figure, XYZ Company has only four registered IP addresses. One address is taken by the perimeter router, the PIX Firewall, and bastion host.

The example configuration is as follows:

```
ip address (inside) 10.0.0.1 255.255.255.0
ip address (outside) 192.168.0.2 255.255.255.0
route (outside) 0.0.0.0 0.0.0.0 192.168.0.1
```

IP addresses are assigned to the internal and external interfaces. A single registered IP address is put into the global pool, and is shared by all outgoing access for network 10.0.0.0:

```
global (outside) 1 192.168.0.9 netmask 255.255.255.0
nat (inside) 1 10.0.0.0 255.0.0.0
```

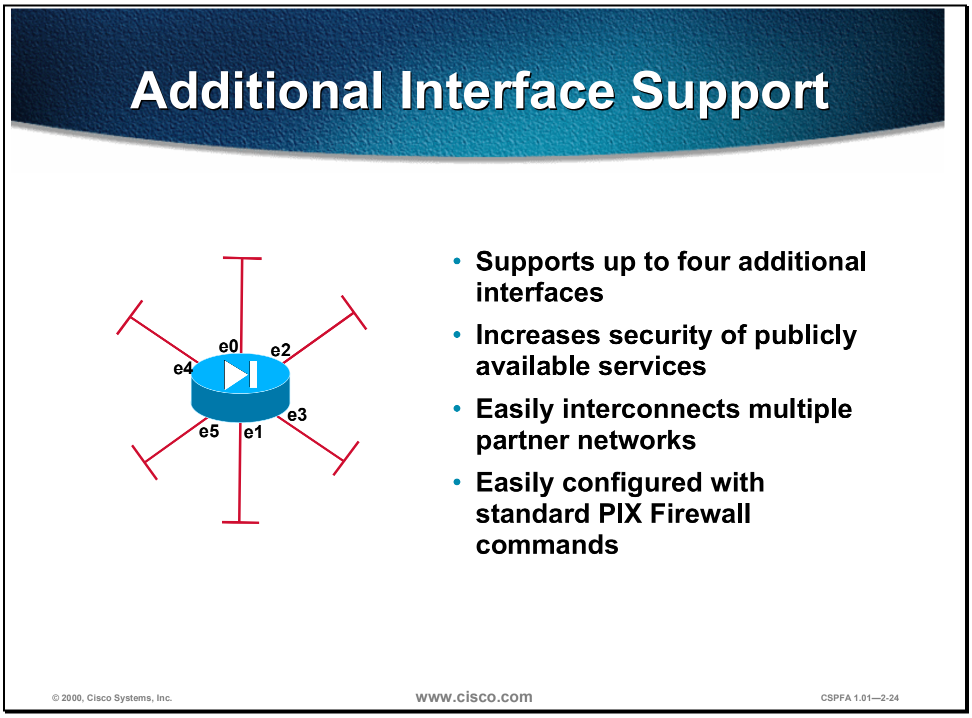


Another feature to control outbound connections is the ability to control which internal IP addresses are visible on the outside. The **nat 0** command lets you disable address translation so that inside IP addresses are visible on the outside without address translation. Use this feature when you have Network Information Center- (NIC) registered IP addresses on your inside network that you want to be accessible on the outside network. Use of **nat 0** depends on your security policy. If your policy allows for internal clients to have their IP address exposed to the Internet, then **nat 0** is the process to provide that service.

In the figure above, the address 192.168.1.9 is not translated. When you enter **`nat (inside) 0 192.168.1.9 255.255.255.255`**, the PIX Firewall displays the following message: `nat 0 192.168.1.9`.

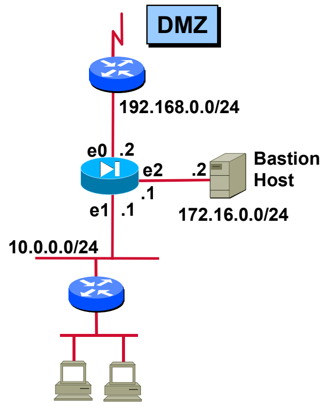
Multiple Interface Configurations

This section discusses the configuration of multiple interfaces to the PIX Firewall.



The PIX Firewall supports up to four additional perimeter interfaces for platform extensibility and security policy enforcement on publicly accessible services. The multiple perimeter interfaces enable the PIX Firewall to protect publicly accessible Internet, mail, and Domain Name System (DNS servers on the DMZ. Web-based and traditional electronic data interchange (EDI) applications that link vendors and customers are also more secure and scalable when implemented using a physically separate network. As the trend toward building these extranet or partnernet applications accelerates, the PIX Firewall is already prepared to accommodate them.

Three-Interface Configuration

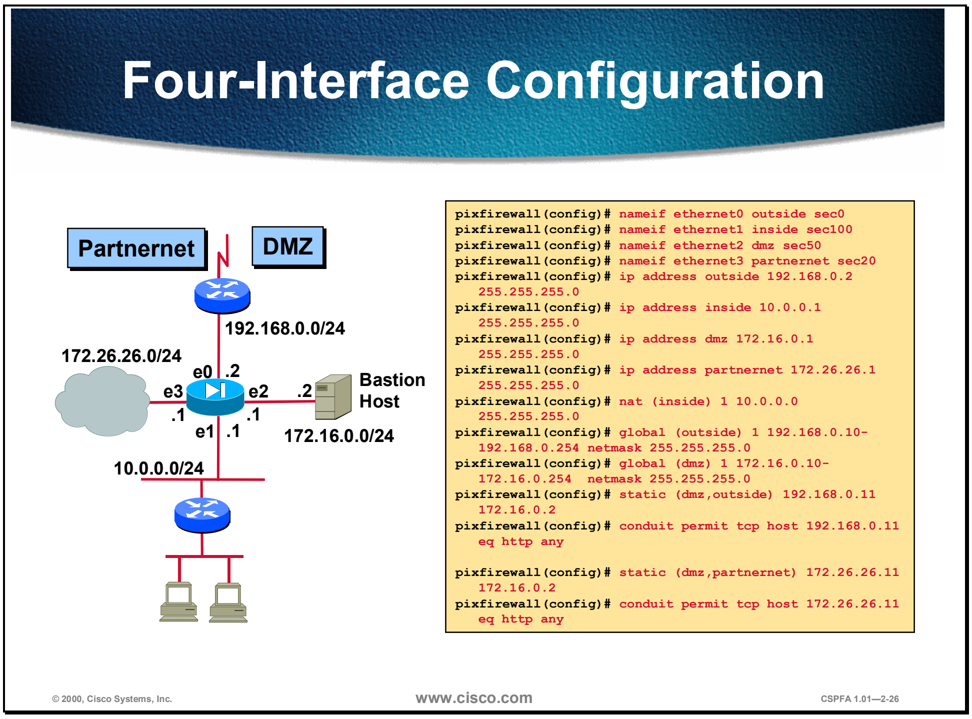



```
pixfirewall(config)# nameif ethernet0 outside
   sec0
pixfirewall(config)# nameif ethernet1 inside
   sec100
pixfirewall(config)# nameif ethernet2 dmz sec50

pixfirewall(config)# ip address outside
   192.168.0.2 255.255.255.0
pixfirewall(config)# ip address inside 10.0.0.1
   255.255.255.0
pixfirewall(config)# ip address dmz 172.16.0.1
   255.255.255.0

pixfirewall(config)# nat (inside) 1 10.0.0.0
   255.255.255.0
pixfirewall(config)# global (outside) 1
   192.168.0.10-192.168.0.254 netmask
   255.255.255.0
pixfirewall(config)# global (dmz) 1
   172.16.0.10-172.16.0.254  netmask
   255.255.255.0

pixfirewall(config)# static (dmz,outside)
   192.168.0.11 172.16.0.2
pixfirewall(config)# conduit permit tcp host
   192.168.0.11 eq http any
```

 www.cisco.com CSPFA 1.01—2-25

A third interface is configured as shown in the figure above. When your PIX Firewall is equipped with three or more interfaces, use the following guidelines to configure it while employing NAT:

- The outside interface cannot be renamed or given a different security level.
- An interface is always "outside" with respect to another interface that has a higher security level. Packets cannot flow between interfaces that have the same security level.
- Use a single default route statement to the outside interface only. Set the default route with the **route** command.
- Use the **nat** command to let users on the respective interfaces start outbound connections. Associate the ***nat_id*** with the ***global_id*** in the **global** command statement. The valid ID numbers can be any positive number up to two billion.
- After you have cmpleted a configuration in which you add, change, or remove a **global** statement, save the configuration, and enter the **clear xlate** command so that the IP addresses will be updated in the translation table.
- To permit access to servers on protected networks, use the **static** and **conduit** commands.



In the figure above, the PIX Firewall has four interfaces. Users on all interfaces have access to all servers and hosts (inside, outside, DMZ, and partnernet).

Configuring four interfaces requires more attention to detail, but they are still configured with standard PIX Firewall commands. Enable users on an interface with a higher security level to access hosts on an interface with a lower security level by using the **nat** and **global** commands. For example, enable the inside interface to access the web server on the DMZ interface.

To let users on an interface with a lower security level (users on the partnernet interfaces access the DMZ) to access hosts on an interface with a higher security level, use the **static** and **conduit** commands. As seen in the figure above, the partnernet has a security level of 40 and the DMZ has a security level of 50. The DMZ will use **nat** and **global** commands to speak with the partnernet, and will use statics and conduits to receive traffic from the partnernet.

The table is a quick reference guide for when to use the **nat** or **static** command when configuring varied interfaces in the PIX Firewall.

From This Interface	To This Interface	Use This Command
Inside	Outside	**nat**
Inside	DMZ	**nat**
Inside	Partnernet	**nat**
DMZ	Outside	**nat**
DMZ	Partnernet	**static**
DMZ	Inside	**static**
Partnernet	Outside	**nat**
Partnernet	DMZ	**nat**
Partnernet	Inside	**static**
Outside	DMZ	**static**
Outside	Partnernet	**static**
Outside	Inside	**static**

Lab Exercise: Configure the PIX Firewall

Complete the following lab exercise to practice what you learned in this chapter.

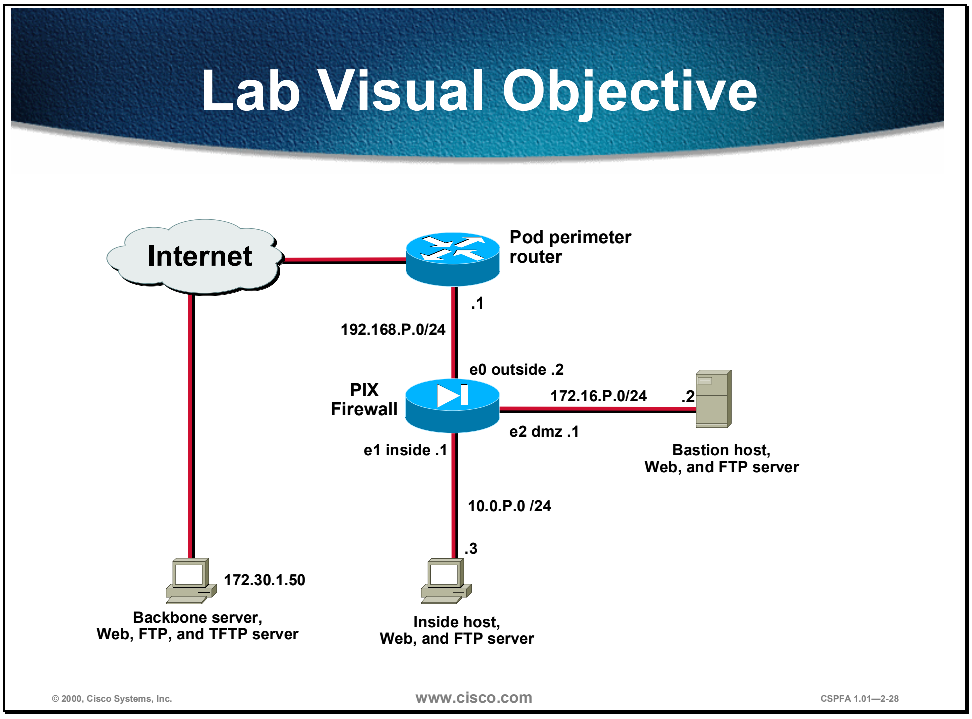
Objectives

In this lab exercise you will complete the following tasks:

- Configure PIX Firewall interfaces.
- Configure global addresses.
- Test the inside, outside, and DMZ interface connectivity.
- Configure globals and NAT.
- Test global and NAT configuration.
- Configure a static and conduit from the PIX Firewall outside interface to the Windows NT server inside the network.
- Configure multiple inside interfaces.
- Configure outside access to the DMZ.

Visual Objectives

The following figure displays the configuration you will complete in this lab exercise.



Setup

Before starting this lab exercise, access the PIX Firewall console port using a HyperTerminal connection.

Directions

You will assign IP addresses and review all entries. Substitute your pod number wherever you see the letter *P*.

Perform the following steps in this lab exercise:

- Configure the PIX Firewall interfaces.
- Test the inside, outside, and DMZ interface connectivity.
- Configure global addresses, NAT, and routing for inside and outside interfaces.

Task 1: Configure PIX Firewall Interfaces

To configure PIX Firewall Ethernet interfaces, complete the following steps:

Step 1 Change to privileged mode:

```
pixfirewall> enable
```

Step 2 When prompted for the password, leave blank and press **Enter**.

Step 3 Enter the **configure terminal** command to enter into configuration mode:

```
pixfirewall> config terminal
```

Step 4 Assign a hostname to your PIX Firewall:

```
pixfirewall(config)# hostname pixP
```

(where *P* = pod number)

Step 5 Assign the PIX Firewall DMZ interface a name (dmz) and security level (50):

```
pixfirewall(config)# nameif e2 dmz security50
pixfirewall(config)# show nameif
nameif ethernet0 outside security0
nameif ethernet1 inside security100
nameif ethernet2 dmz security50
```

Step 6 Enable the Ethernet 0, Ethernet 1, and Ethernet 2 interfaces for an Intel 100 full interface card.

Note By default the interfaces are disabled. You must enable all interfaces you intend to use.

```
pixfirewall(config)# interface e0 100full
pixfirewall(config)# interface e1 100full
pixfirewall(config)# interface e2 100full
pixfirewall(config)# show interface
interface ethernet0 "outside" is up, line protocol is up
  Hardware is i82558 ethernet, address is 0090.2724.fd0f
  IP address 127.0.0.1, subnet mask 255.255.255.255
  MTU 1500 bytes, BW 10000 Kbit full duplex
        0 packets input, 0 bytes, 0 no buffer
        Received 0 broadcasts, 0 runts, 0 giants
        0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
        0 packets output, 0 bytes, 0 underruns
interface ethernet1 "inside" is up, line protocol is up
  Hardware is i82558 ethernet, address is 0090.2716.43dd
  IP address 127.0.0.1, subnet mask 255.255.255.255
  MTU 1500 bytes, BW 100000 Kbit full duplex
        184 packets input, 15043 bytes, 0 no buffer
        Received 179 broadcasts, 0 runts, 0 giants
        0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
        0 packets output, 0 bytes, 0 underruns
interface ethernet2 "dmz" is up, line protocol is up
  Hardware is i82558 ethernet, address is 0090.2725.060d
```



```
  IP address 127.0.0.1, subnet mask 255.255.255.255
  MTU 1500 bytes, BW 10000 Kbit full duplex
        0 packets input, 0 bytes, 0 no buffer
        Received 0 broadcasts, 0 runts, 0 giants
        0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
        0 packets output, 0 bytes, 0 underruns
```

Step 7 Assign IP addresses to the inside, outside, and DMZ network interface cards:

```
pixfirewall(config)# ip address outside 192.168.P.2 255.255.255.0
pixfirewall(config)# ip address inside 10.0.P.1 255.255.255.0
pixfirewall(config)# ip address dmz 172.16.P.1 255.255.255.0
```

(where *P* = pod number)

Step 8 Ensure that the IP addresses are correctly configured and are associated with the proper network interface:

```
pixfirewall(config)# show ip address
System IP Addresses:
        ip address outside 192.168.P.2 255.255.255.0
        ip address inside 10.0.P.1 255.255.255.0
        ip address dmz 172.16.P.1 255.255.255.0Current IP Addresses:
        ip address outside 192.168.P.2 255.255.255.0
        ip address inside 10.0.P.1 255.255.255.0
        ip address dmz 172.16.P.1 255.255.255.0
```

Step 9 Write the configuration to the Flash memory:

```
pixfirewall(config)# write memory
Building configuration...
Cryptochecksum: d4d9ae69 9f7c734c babeef58 54b69c91
```

Task 2: Configure Global Addresses

To configure a global address pool, NAT, and routing, complete the following steps:

Step 1 Assign one pool of NIC-registered IP addresses for use by outbound connections:

```
pixfirewall# config terminal
pixfirewall(config)# global (outside) 1 192.168.P.10-192.168.P.254 netmask
255.255.255.0
pixfirewall(config)# show global
global (outside) 1 192.168.P.10-192.168.P.254 netmask 255.255.255.0
```

(where *P* = pod number)

Step 2 Configure the PIX Firewall to allow all inside hosts to use NAT for outbound access:

```
pixfirewall(config)# nat (inside) 1 0 0
```

Step 3 Display the currently configured NAT:

```
pixfirewall(config)# show nat
nat (inside) 1 0.0.0.0 0.0.0.0 0 0
```

Step 4 Assign a default route:

```
pixfirewall(config)# route outside 0 0 192.168.P.1
```

Step 5 Display currently configured routes:

```
pixfirewall(config)# show route
outside 0.0.0.0 0.0.0.0 192.168.P.1 1 OTHER static
```

Step 6 Write the current configuration to Flash memory:

```
pixfirewall(config)# write memory
```

Task 3: Test the Inside, Outside, and DMZ Interface Connectivity

To test and troubleshoot interface connectivity using the PIX Firewall **ping** command, complete the following steps:

Step 1 Ping the inside interface:

```
pixfirewall# ping inside 10.0.P.1
10.0.P.1 response received -- 10ms
10.0.P.1 response received -- 10ms
10.0.P.1 response received -- 10ms
```

(where *P* = pod number)

Step 2 Ping your inside host:

```
pixfirewall# ping inside 10.0.P.3
10.0.P.3 response received -- 10ms
10.0.P.3 response received -- 10ms
10.0.P.3 response received -- 10ms
```

(where *P* = pod number)

Step 3 Ping the outside interface:

```
pixfirewall# ping outside 192.168.P.2
192.168.P.2 response received -- 10ms
192.168.P.2 response received -- 10ms
192.168.P.2 response received -- 10ms
```

(where *P* = pod number)

Step 4 Ping your pod perimeter router:

```
pixfirewall# ping outside 192.168.P.1
192.168.P.1 response received -- 10ms
192.168.P.1 response received -- 10ms
192.168.P.1 response received -- 10ms
```

(where *P* = pod number)

Step 5 Ping the DMZ interface:

```
pixfirewall# ping dmz 172.16.P.1
172.16.P.1 response received -- 10ms
172.16.P.1 response received -- 10ms
172.16.P.1 response received -- 10ms
```

(where *P* = pod number)

Step 6 Ping your bastion host:

```
pixfirewall# ping dmz 172.16.P.2
172.16.P.2 response received -- 10ms
172.16.P.2 response received -- 10ms
172.16.P.2 response received -- 10ms
```

(where *P* = pod number)

Task 4: Configure Global and NAT

Enter the following commands to configure PIX Firewall global address pools and routing:

Step 1 Remove NAT:

```
pixfirewall(config)# no nat (inside) 1 0 0
```

Step 2 Configure NAT for the internal network range of IP addresses

```
pixfirewall(config)# nat (inside) 1 10.0.P.0 255.255.255.0 0 0
Display currently configured NAT
pixfirewall(config)# show nat
nat (inside)1 10.0.P.0 255.255.255.0 0 0
```

(where *P* = pod number)

Step 3 Allow ICMP and ping packets through the PIX Firewall:

```
pixfirewall(config)# conduit permit icmp any any
```

Step 4 Display the currently configured conduit:

```
pixfirewall(config)# show conduit
```

Step 5 Write the current configuration to Flash memory:

```
pixfirewall(config)# write memory
```

Step 6 Write the current configuration to the terminal:

```
pixfirewall(config)# write terminal
```

Step 7 Use the **clear xlate** command after configuring with the **nat** and **global** commands to make the global IP addresses available in the translation table:

```
pixfirewall(config)# clear xlate
pixfirewall(config)# show xlate
```

Task 5: Test Globals and NAT Configuration

To test the globals and NAT configuration, you must complete the following:

Step 1 From your Windows command line, ping the perimeter router:

```
C:\> ping 192.168.P.1
```

(where *P* = pod number)

Step 2 Test the operation of the global and NAT you configured by originating connections through the PIX Firewall:

1. Open a web browser on the Windows NT server.

2. Use the web browser to access the IP address perimeter router backbone server and open the server's HTTP server: **http://172.30.1.50**. Stop the connection and then initiate a new connection.

Step 3 Observe the translation table with the **show xlate** command:

```
pixfirewall(config)# show xlate
```

Your display should appear similar to the following:

```
Global 192.168.P.X Local 10.0.P.3 nconns 1 econns 0 flags -
```

Note how the global addresses have incremented and are chosen from the low end of the global range.

Task 6: Configure a Static and Conduit from the PIX Firewall Outside Interface to the Windows NT Server Inside the Network

Configure a static translation so that traffic originated from the internal Windows NT server always has the same source address on the outside interface of the PIX Firewall. Test the static and conduit by pinging the Windows NT server from the perimeter router. In a production environment, you should remove the **conduit permit icmp any any** command to prevent a potential security breach. Use the following commands:

Step 1 Clear the translation table:

```
pixfirewall(config)# clear xlate
```

Step 2 Create a static translation from the outside PIX Firewall interface to the internal host

```
pixfirewall(config)# static (inside,outside) 192.168.P.10 10.0.P.3
pixfirewall(config)# conduit permit tcp host 192.168.P.10 eq www any
```

(where *P* = pod number)

Step 3 Turn on ICMP monitoring at the PIX Firewall:

```
pixfirewall(config)# debug icmp trace
ICMP trace on Warning: this may cause problems on busy networks
```

Step 4 Ping the perimeter router from your Windows NT server to test the translation. Observe the source and destination of the packets at the console of the R1 perimeter router from each of the following locations

```
C:\> ping 192.168.P.1
```

(where *P* = pod number)

Note the example display for pixfirewall:

```
Outbound ICMP echo request 10.0.P.3 > 192.168.P.10 > 192.168.P.1
Inbound  ICMP echo reply 192.168.P.1 > 192.168.P.10 > 10.0.P.3
Outbound ICMP echo request 10.0.P.3 > 192.168.P.10 > 192.168.P.1
Inbound  ICMP echo reply 192.168.P.1 > 192.168.P.10 > 10.0.P.3
Outbound ICMP echo request 10.0.P.3 > 192.168.P.10 > 192.168.P.1
Inbound  ICMP echo reply 192.168.P.1 > 192.168.P.10 > 10.0.P.3
Outbound ICMP echo request 10.0.P.3 > 192.168.P.10 > 192.168.P.1
```



```
Inbound  ICMP echo reply 192.168.P.1 > 192.168.P.10 > 10.0.P.3
```

(where *P* = pod number)

Observe the source, destination, and translated addresses on the PIX Firewall console.

Step 5 Use the web browser to access the IP address of the peer's host by entering **http://192.168.Q.10** (where *Q* = peer pod number) in your web browser..

Step 6 Ping a peer inside host from your inside host as allowed by the conduit via the static.

```
C:\> ping 192.168.Q.10
```

(where *Q* = peer pod number)

Step 7 Turn off ICMP monitoring at the PIX Firewall.

```
pixfirewall(config)# no debug icmp trace
```

Task 7: Configure Inside Multiple Interfaces

Configure the PIX Firewall to allow access to the DMZ from the inside and outside network. Complete the following steps to configure the global address pools, NAT, and routing for the DMZ interface:

Step 1 Assign one pool of IP addresses for hosts on the public DMZ:

```
pixfirewall(config)# global (dmz) 1 172.16.P.10-172.16.P.254 netmask 255.255.255.0
```

(where *P* = pod number)

Step 2 Name the bastion host using the **name** command. The name configured here will be used in a later lab step:

```
pixfirewall(config)# name 172.16.P.2 bastionhost
pixfirewall(config)# show name
name 172.16.P.2 bastionhost
```

(where *P* = pod number)

Step 3 Clear the translation table so that the global IP address will be updated in the table:

```
pixfirewall(config)# clear xlate
```

Step 4 Write the current configuration to Flash memory:

```
pixfirewall(config)# write memory
```

Step 5 Test connectivity to the bastion host from your internal host.

```
C:\> ping 172.16.P.2
```

(where *P* = pod number)

Step 6 Test web access to your bastion host from the Windows NT server by doing the following:

1. Open a web browser on the Windows NT server.
2. Use the web browser to access the IP address of your bastion host: **http://172.16.P.2**.. The home page of the bastion host should appear on your web browser. (where *P* =pod number)

3. Use the **show arp**, **show conn**, and **show xlate** commands to observe the transaction:

```
pixfirewall(config)# show arp
        outside 192.168.P.1 00e0.1e41.8762
        inside 10.0.P.3 00e0.b05a.d509
        dmz bastionhost 00e0.1eb1.78df
pixfirewall(config)# show xlate
Global 172.16.P.2 Local 10.0.P.10 static nconns 0 econns 0 flags s
Global 192.168.P.3 Local 10.0.P.10 nconns 0 econns 0 flags -
pixfirewall(config)# show conn
0 in use, 3 most used
```

Step 7 Test FTP access to the bastion host from your Windows NT server by doing the following:

1. Establish an FTP session to the bastion host: **Start>Run>ftp 172.16.P.2**. You have reached the bastion host if you receive the message "**Connected to 172.16.P.2**"(where *P* = pod number).
2. Quit the FTP session if you were able to connect, and log in: **ftp> quit**.

Task 8: Configure Outside Access to the DMZ

Configure the PIX Firewall to permit outside access to hosts in the DMZ. Configure a static and conduit to test communications using ping between perimeter routers and the bastion host. Then configure HTTP and FTP access. Complete the following steps:

Step 1 Create a static translation from the outside interface to the bastion host on the DMZ interface:

```
pixfirewall(config)# static (dmz,outside) 192.168.P.11 bastionhost
```

(where *P* = pod number)

Step 2 Configure a conduit to allow pings from perimeter routers to the static assigned to the DMZ bastion host:

```
pixfirewall(config)# conduit permit icmp host 192.168.P.11 any
```

(where *P* = pod number)

Step 3 Ping a peer bastion host from your internal host as allowed by the conduit via the static

```
C:\> ping 192.168.Q.11
```

(where *Q* = peer pod number)

Step 4 View current static translations:

```
pixfirewall(config)# show xlate
Global 192.168.P.11 Local 10.0.P.3 static nconns1 econns1
Global 192.168.P.11 Local bastionhost static nconns0 econns0
```


Step 5 Configure conduits to allow web and FTP access to the bastion host from the outside and then test the conduits. Configure the conduits to allow TCP traffic from clients on the outside network to access the DMZ bastion host using the previously configured static:

```
pixfirewall(config)# conduit permit tcp host 192.168.P.11 eq www  any
pixfirewall(config)# conduit permit tcp host 192.168.P.11 eq ftp  any
```

(where *P* = pod number)

Step 6 Display the conduits that you have just configured:

```
pixfirewall(config)# show conduit
conduit permit tcp host 192.168.1.11 eq www any (hitcnt=0)
conduit permit tcp host 192.168.1.11 eq ftp any (hitcnt=0)
```

Step 7 Test web access to the bastion hosts of opposite pod groups by doing the following:

1. Open a web browser on the client PC.
2. Identify another pod group that is ready for a test.
3. Use the web browser to access the IP address of the static mapped to the bastion host of the opposite pod group: **http://192.168.Q.11**
4. Have an opposite pod group test your static and conduit configuration.
5. Use the **show arp**, **show conn**, and **show xlate** commands to observe the transaction.

Step 8 Test FTP access to the bastion hosts of other pod groups by doing the following:

1. Identify another pod group that is ready for a test.
2. On your client PC, use FTP to get into the bastion host of another pod group. **Start>Run>ftp 192.168.Q.11**

   (where *Q* = peer pod number)
3. Have an opposite pod group use FTP to get into your bastion host to test your static and conduit configuration.
4. Use the **show arp**, **show conn**, and **show xlate** commands to observe the transaction.

Step 9 Write the current configuration to the terminal and verify that you have entered the previous commands correctly. Your configuration should appear similar to the following:

```
pixfirewall(config)# write terminal
Building configuration...
Building configuration...
: Saved
:
PIX Version 5.0(1)
nameif ethernet0 outside security0
nameif ethernet1 inside security100
nameif ethernet2 dmz security50
enable password 8Ry2YjIyt7RRXU24 encrypted
passwd 2KFQnbNIdI.2KYOU encrypted
hostname pixfirewall
```

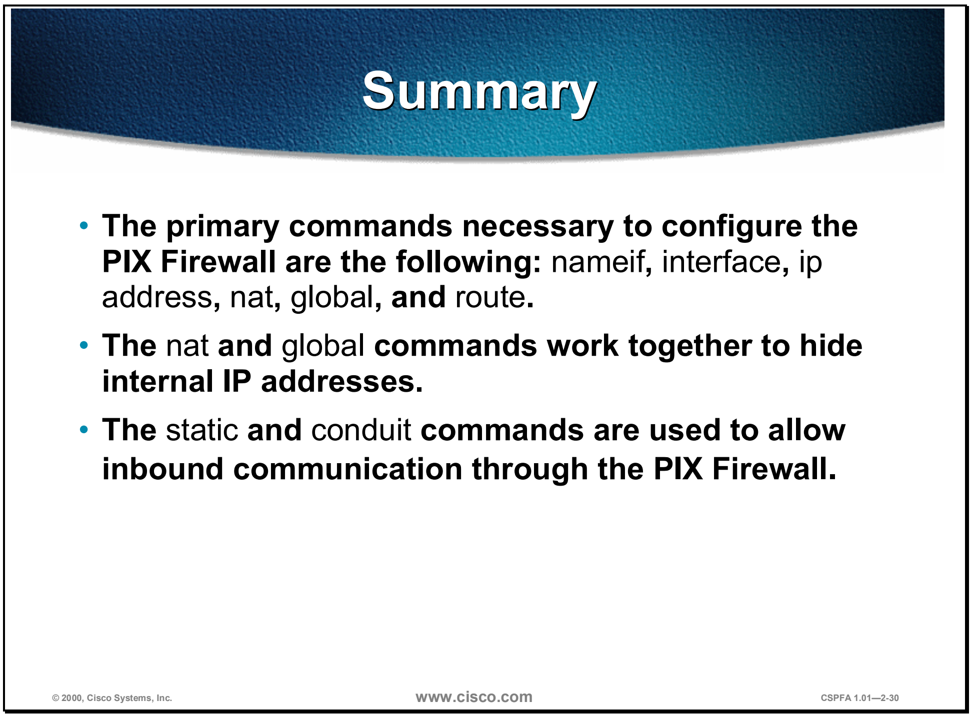
```
fixup protocol ftp 21
fixup protocol http 80
fixup protocol smtp 25
fixup protocol h323 1720
fixup protocol rsh 514
fixup protocol sqlnet 1521
names
pager lines 24
no logging timestamp
no logging standby
no logging console
no logging monitor
no logging buffered
no logging trap
logging facility 20
logging queue 512
interface ethernet0 auto
interface ethernet1 auto
interface ethernet2 auto
mtu outside 1500
mtu inside 1500
mtu dmz 1500
ip address outside 192.168.P.2 255.255.255.0
ip address inside 10.0.P.1 255.255.255.0
ip address dmz 172.16.P.1 255.255.255.0
no failover
failover timeout 0:00:00
failover ip address outside 0.0.0.0
failover ip address inside 0.0.0.0
failover ip address dmz 0.0.0.0
arp timeout 14400
global (outside) 1 172.16.21.10-172.16.21.254 netmask 255.255.255.0
nat (inside) 1 0.0.0.0 0.0.0.0 0 0
no rip outside passive
no rip outside default
no rip inside passive
no rip inside default
no rip dmz passive
no rip dmz default
route outside 0.0.0.0 0.0.0.0 192.168.P.1 1
timeout xlate 3:00:00 conn 1:00:00 half-closed 0:10:00 udp 0:02:00
timeout rpc 0:10:00 h323 0:05:00
timeout uauth 0:05:00 absolute
aaa-server TACACS+ protocol tacacs+
aaa-server RADIUS protocol radius
no snmp-server location
no snmp-server contact
snmp-server community public
no snmp-server enable traps
telnet timeout 5
terminal width 80
Cryptochecksum:9963c491006b1296815f3437947fab81
: end
```

Step 10 Write the current configuration to Flash memory:

```
pixfirewall(config)# write memory
Building configuration...
Cryptochecksum: ae9fc9fc a3005950 f9daec62 5683c88e
[OK]
```

Summary

This section summarizes what you learned in this chapter.



3

Access Control Configuration and Content Filtering

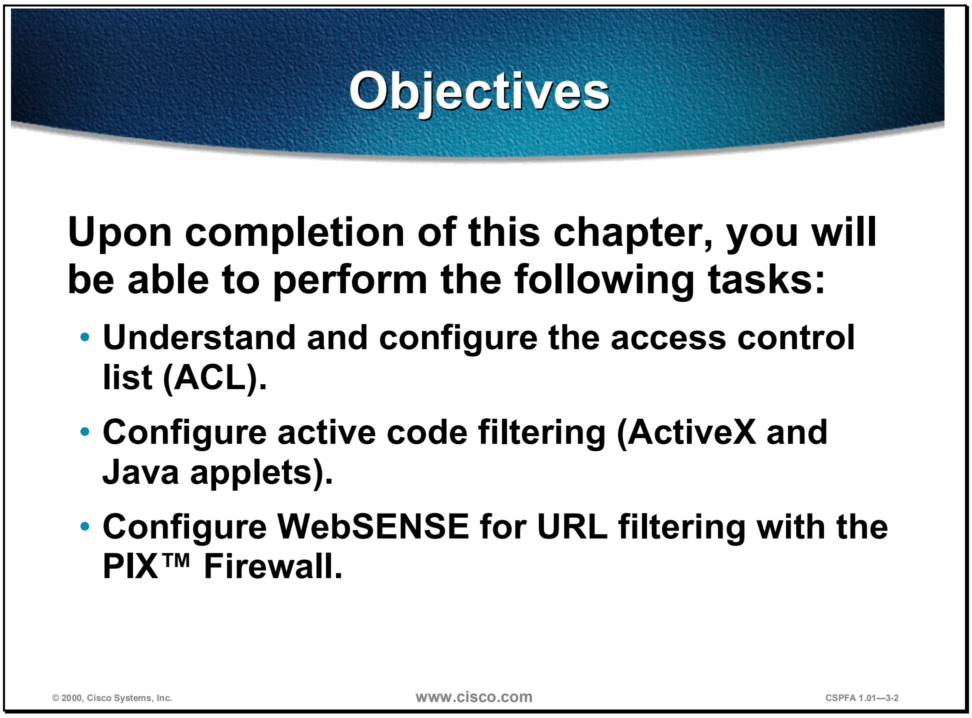
Overview

This chapter includes the following topics:

- Objectives
- Access control through the PIX Firewall
- Malicious active code filtering
- URL filtering with WebSENSE
- Lab exercise
- Summary

Objectives

This section lists the chapter's objectives.



Access Control Through the PIX Firewall

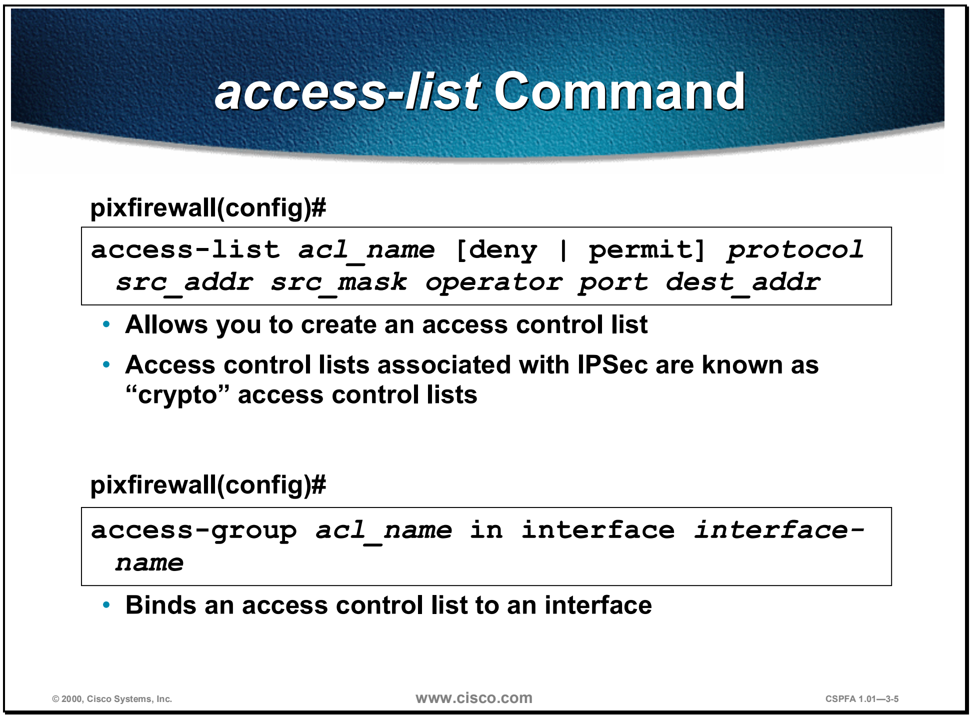
This section discusses access control through the PIX™ Firewall using an access control list (ACL).

An ACL is a list kept by routers and the PIX Firewall to control access to and from the router or firewall (for example, to prevent packets with a certain IP address from leaving a particular interface). An ACL is implemented using two commands: the **access-list** command and the **access-group** command.

Use the **access-list** command to create an ACL. The **access-group** command binds the ACL to a specific interface on the router or PIX Firewall. Only one ACL can be bound to an interface using the **access-group** command.

The **access-list** and **access-group** commands are an alternative to the **outbound** command statement. The **access-list** and **access-group** commands also take precedence over the **outbound** command statement.

Note Cisco recommends using the **access-list** and **access-group** commands for ACLs instead of the **outbound** command because the **outbound** command is a PIX Firewall-specific command and Cisco is moving toward commands that are based on the Cisco IOS™.



The **access-list** command uses the same syntax as the Cisco IOS **software access-list** command except that the subnet mask in the PIX Firewall **access-list** command is reversed from the Cisco IOS software version of this command. For example, a subnet mask specified as 0.0.0.255 in the Cisco IOS **access-list** command would be specified as 255.0.0.0 in the PIX Firewall **access-list** command. The syntax for the **access-list** command is as follows:

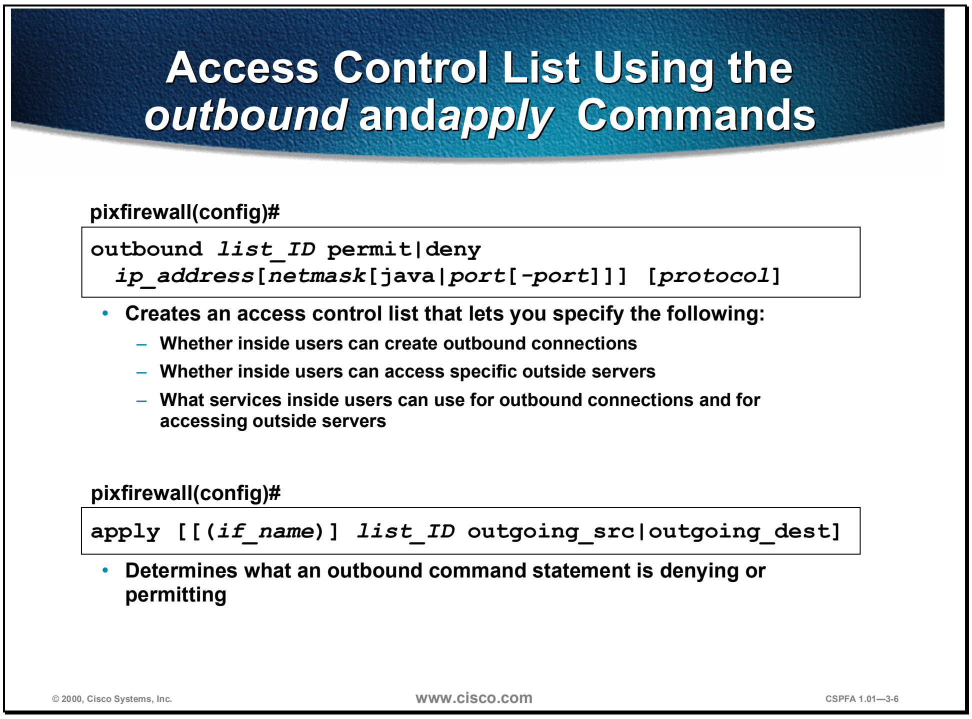
access-list ***acl_name*** **[deny | permit]** ***protocol src_addr src_mask operator port dest_addr***

Argument	Description
acl_name	Name of an ACL.
deny	Does not allow a packet to travel through the PIX Firewall. By default, the PIX Firewall denies all inbound packets unless you specifically permit access.
permit	Selects a packet to travel through the PIX Firewall.
protocol	Name or number of an IP protocol.
src_addr	Address of the network or host from which the packet is being sent.
src_mask	Netmask address of the source address.
operator	Specifies a port or a port range.
port	Services you permit to be used while accessing ***src_addr*** or ***dest_addr***.
dest_addr	IP address of the network or host to which the packet is being sent.

The syntax for the **access group** command is as follows:

access-group ***acl_name*** **in interface** ***interface-name***

Argument	Description
acl_name	The name associated with a given ACL.
in interface	Filters on inbound packets at the given interface.
interface-name	The name of the network interface.



The **outbound** command creates an access control list (ACL) that enables you to specify the following:

- Whether inside users can use outbound connections
- Whether inside users can access outside servers
- What services inside users can use for outbound connections and for accessing outside servers
- Whether outbound connections can execute java applets on the inside network

Outbound lists are filters on outgoing packets from the PIX Firewall. The filter can be based on the source IP address, the destination IP address, and the destination port or protocol as specified by the rules. The syntax for the **outbound** command is as follows:

outbound ***list_ID*** **permit | deny** ***ip_address*** **[*netmask*[java | *port*[-*port*]]] [*protocol*]**

Argument	Description
list_ID	A tag number for the ACL.
permit	Allows the ACL to access the specified IP address and port.
deny	Denies the ACL access to the specified IP address and port.
Ip_address	The IP address for this ACL entry.
netmask	The network mask for comparing with the IP address.
java	Indicates port 80.

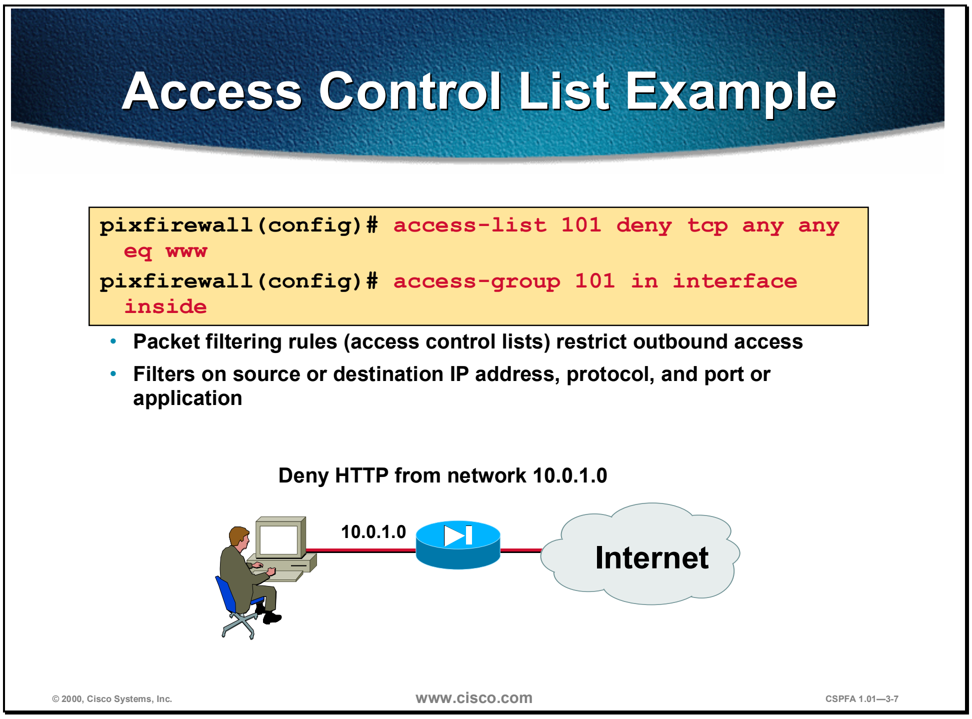
Argument	Description
port	A port or range of ports that the ACL is permitted or denied access to.
protocol	Limit outbound access to *udp*, *tcp*, or *icmp* protocols.

The use of an **outbound** command requires use of the **apply** command. The **apply** command lets you specify whether the ACL applies to inside users' ability to start outbound connections with the **apply** command ***outgoing_src*** option, or whether the ACL applies to the inside users' ability to access servers on the outside network with the **apply** commans ***outgoing_dest*** option. The syntax for the **apply** command is as follows:

apply [[(*if_name*)] *list_ID outgoing_src* | *outgoing_dest*]

Argument	Description
if_name	The network interface originating the connection.
list_ID	A tag number for the ACL.
outgoing_src	Denies or permits an internal IP address to start outbound connections using the services specified in the **outbound** command.
outgoing_dest	Denies or permits access to an external IP address using the services specified in the **outbound** command.

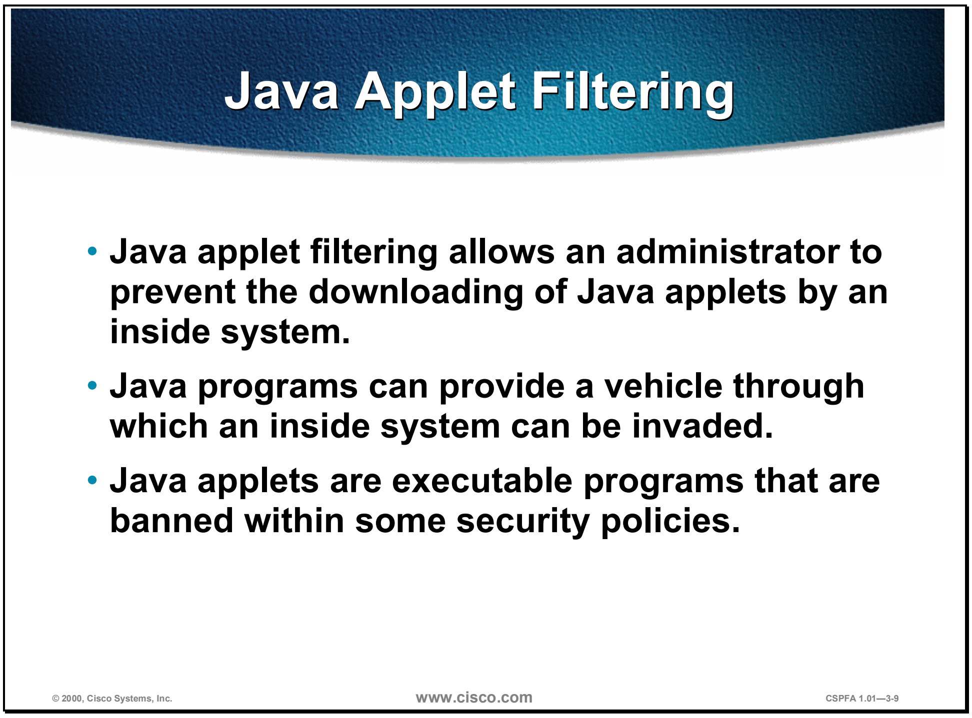
Note After adding, removing, or changing outbound command statements, use the **clear xlate** command.



In the figure above, the PIX Firewall denies HTTP connections from an internal network, but lets all other traffic through.

Malicious Active Code Filtering

The PIX Firewall can filter malicious active codes. Malicious active codes can be used in such applications as Java and ActiveX.



The PIX Firewall supports a Java applet filter that can stop potentially dangerous Java applications on a per-client or per-IP address basis. The **outbound** command with the ***java*** keyword is used to enable filtering of Java applets.

Two problems with this is that some Java applets may be downloaded when you permit access to port 80 (HTTP), and some Java applets can contain hidden code that can destroy data on the internal network. A solution to these problems is to use the **outbound** command to block all Java applets.

Java Applet Filtering Commands

pixfirewall(config)#

```
filter java port[-port] local_ip  mask
  foreign_ip mask
```

- **The** filter java**command filters out Java applets that return to the PIX Firewall from an outbound connection.**
- **Some Java applets can contain malicious code that can manipulate data on the internal network.**
- **Use the** outbound **and** apply **commands to block Java applets.**

 www.cisco.com CSPFA 1.01—3-10

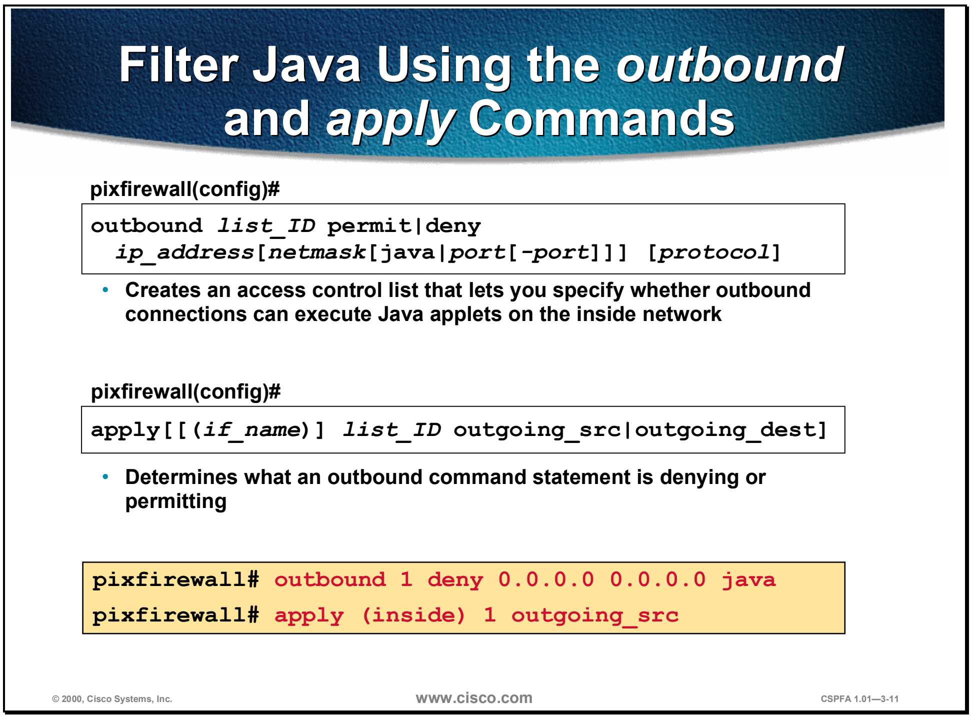
Java filtering lets an administrator prevent Java applets from being downloaded by an inside system. Java applets are executable programs that are banned by many site security policies. The syntax for the **filter java** command is as follows:

filter java *port*[-*port*] *local_ip mask foreign_ip mask*

Argument	Description
***port*[-*port*]**	One or more ports on which Java applets may be received.
local_ip	The IP address interface with the highest security level from which access is sought.
mask	Wildcard mask.
foreign_ip	The IP address of the interface with the lowest security level to which access is sought.

Java programs can provide a vehicle through which an inside system can be invaded or compromised. When Java filtering is enabled, the PIX Firewall searches for the programmed "cafe babe" string and, if found, drops the Java applet. A sample Java class code snippet looks like the following:

```
00000000: café babe 003 002d 0099 0900 8345 0098
```

This is another example of filtering java applets using the **outbound** and **apply** command. The syntax for the **outbound** command is as follows:

outbound *list_ID* permit | deny *ip_address* [*netmask*[java | *port*[-*port*]]] [*protocol*]

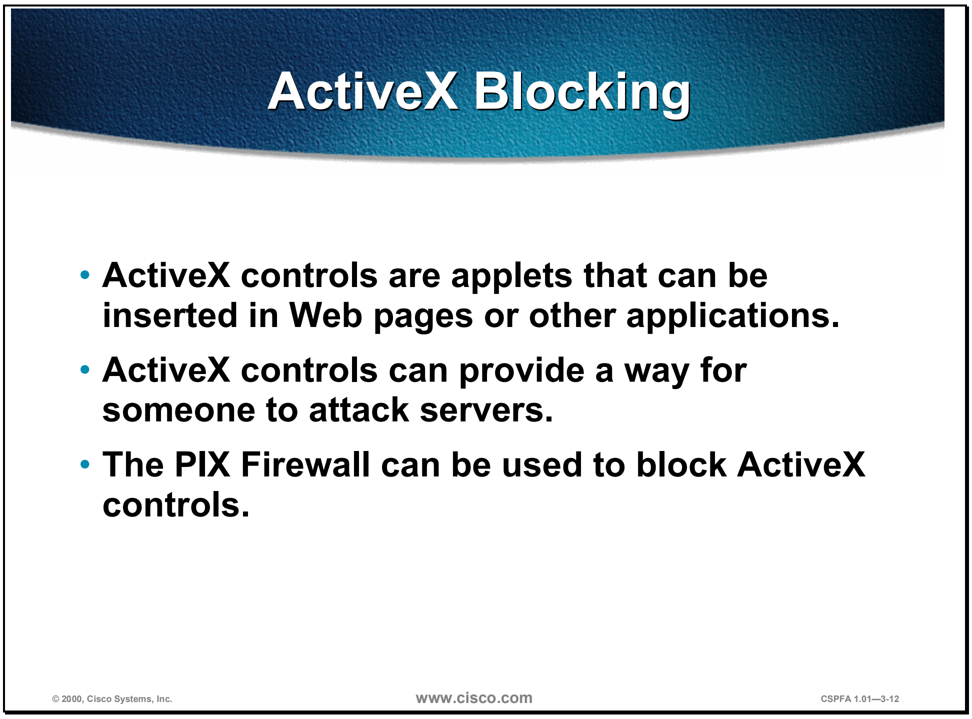
Argument	Description
list_ID	A tag number for the ACL.
permit	Allows the ACL to access the specified IP address and port.
deny	Denies the ACL access to the specified IP address and port.
ip_address	The IP address for this ACL entry.
netmask	The network mask for comparing with the IP address.
java	Indicates port 80.
port	A port or range of ports to which the ACL is permitted or denied access.
protocol	Limit outbound access to the *udp*, *tcp*, or *icmp* protocols.

The syntax for the **apply** command is as follows:

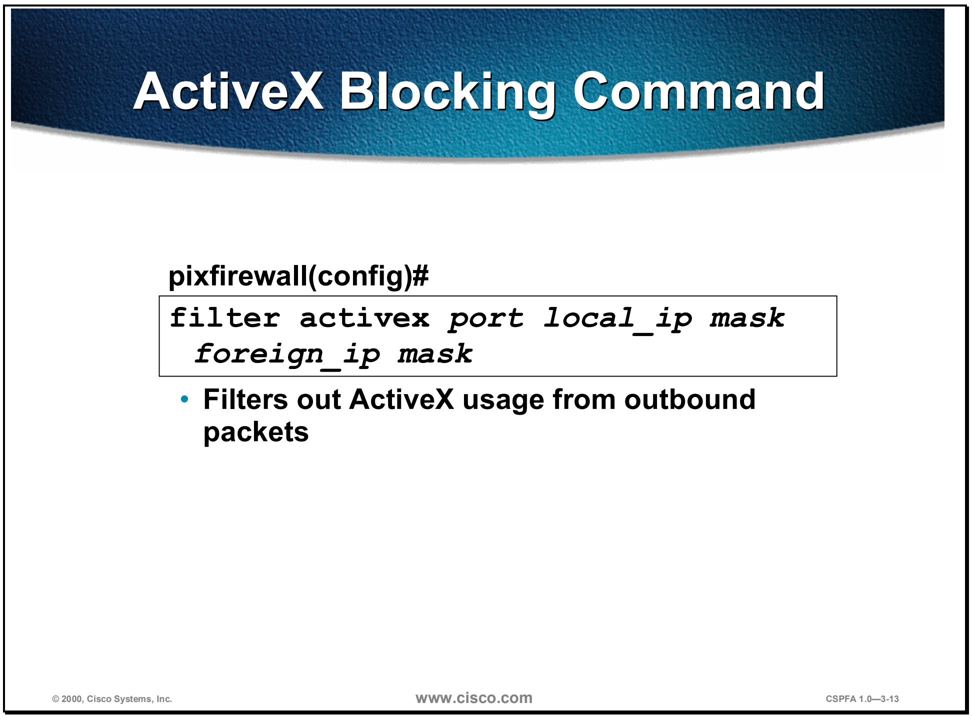
apply [[(*if_name*)] *list_ID outgoing_src* | *outgoing_dest*]

Argument	Description
if_name	The network interface originating the connection.
list_ID	A tag number for the ACL.

Argument	Description
outgoing_src	Denies or permits an internal IP address the ability to start outbound connections using the services specified in the **outbound** command.
outgoing_dest	Denies or permits access to an external IP address using the services specified in the **outbound** command.



ActiveX controls, formerly known as Object Linking and Embedding (OLE) or Object Linking and Embedding control (OCX), are applets that can be inserted in web pages—often used in animations—or in other applications. ActiveX controls create a potential security problem because they can provide a way for someone to attack servers. Because of this potential security problem, you can use the PIX Firewall to block all ActiveX controls.

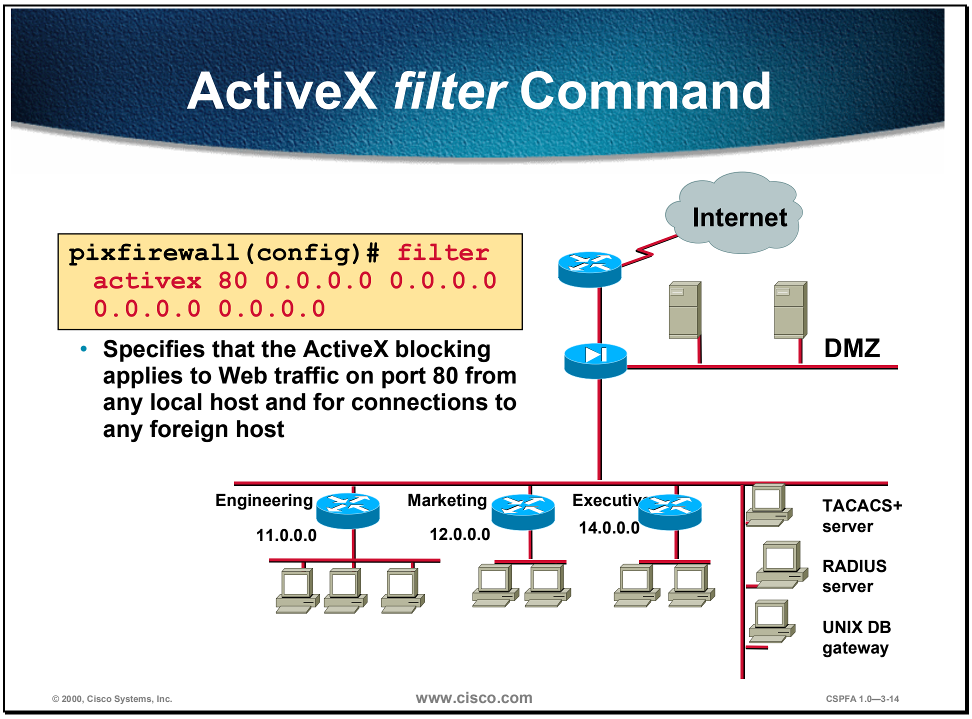


The **filter activex** command filters out ActiveX usage from outbound packets. The syntax for the **filter activex** command is as follows:

filter activex ***port local_ip mask foreign_ip mask***

Argument	Description
activex	Block outbound ActiveX, Java applets, and other HTML <object> tags from outbound packets.
port	The port at which Internet traffic is received on the PIX Firewall.
local_ip	The IP address of the interface with the highest security level from which access is sought.
mask	Wildcard mask.
foreign_ip	The IP address of the interface with the lowest security level to which access is sought.

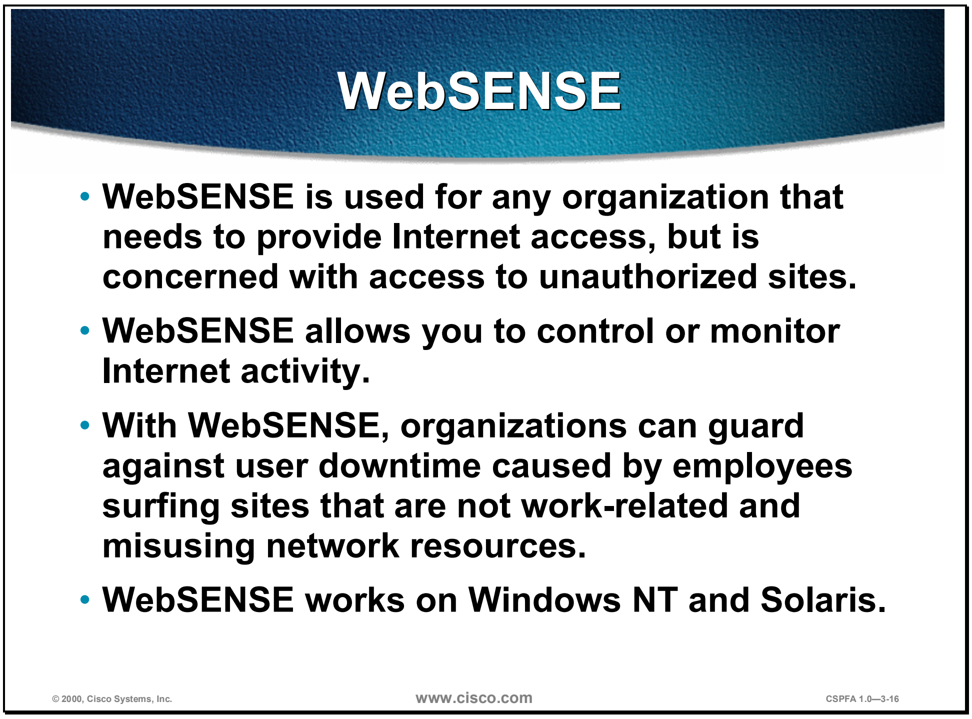
Note ActiveX blocking does not occur when users access an IP address referenced by the **alias** command.



The **filter** command enables or disables outbound URL or HTML filtering. In the figure above, the command specifies that ActiveX is being filtered on port 80 from any internal host and for connection to any external host.

URL Filtering with WebSENSE

This section discusses how to perform URL filtering using WebSENSE.



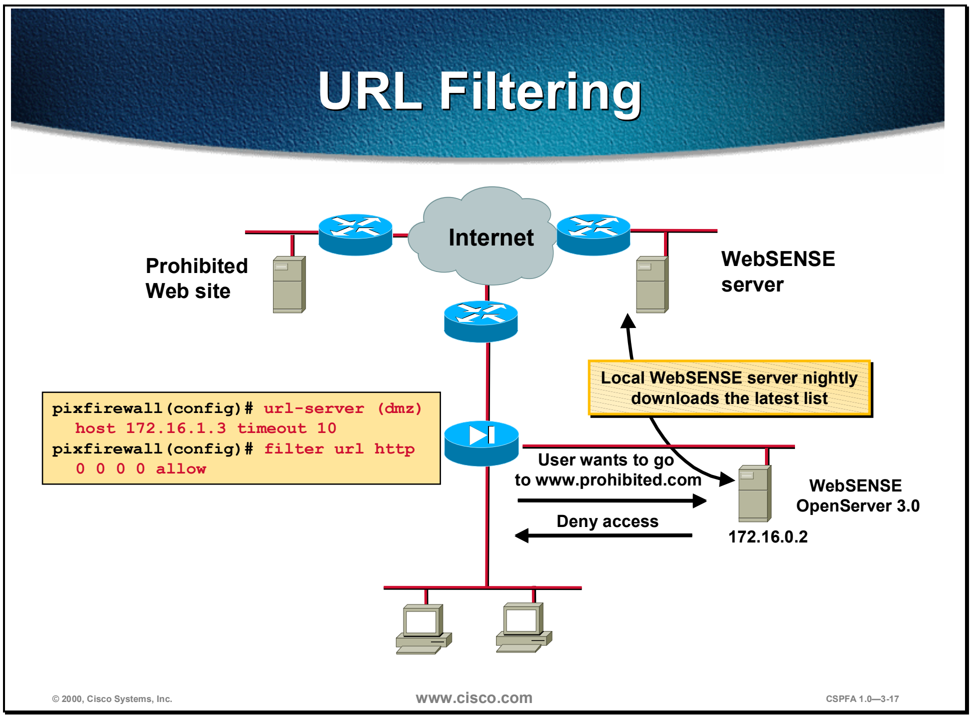
WebSENSE is software that provides integrated, URL filtering for the PIX Firewall, giving network administrators the ability to effectively monitor and control network traffic. WebSENSE is used to block specific URLs because the PIX Firewall cannot. WebSENSE determines whether to block or permit specific URLs based on its configuration and the Master Database. The WebSENSE configuration is the filtering rules you set in WebSENSE. The Master Database is a database of URLs to block. This database is maintained and updated daily by the WebSENSE corporate office. You can choose when, and if you want, to download the URLs from this database.

WebSENSE is useful because between the hours of 9 a.m. and 5 p.m.:

- 30 to 40 percent of Internet surfing is not business related.
- 70 percent of all Internet porn traffic occurs.
- More than 60 percent of online purchases are made.

The following are characteristics of WebSENSE:

- Hosted on Windows NT or Solaris platform
- Located inside or on the perimeter network
- Highly scalable architecture
- Outbound database and lookup processing minimizes impact on firewall performance and security
- Multiple servers can be used to increase capacity
- Has 30 categories of filters
- Is updated daily (adding up to 5,500 sites per week)



When the PIX Firewall receives a request to access a URL from users, it queries WebSENSE to determine whether or not to return the requested URL. WebSENSE checks its configurations and the Master Database to determine whether the URL should be blocked. If the URL should be blocked, WebSENSE displays either the standard blocking message or directs the user requesting the URL to a specified web site.

Designate the WebSENSE Server

pixfirewall(config)#

```
url-server [(if_name)] host ip_address [timeout
  seconds]
```

- The url-server command designates a server that runs WebSENSE.

```
pixfirewall(config)# url-server (inside)
  host 10.0.0.3 timeout 10
```

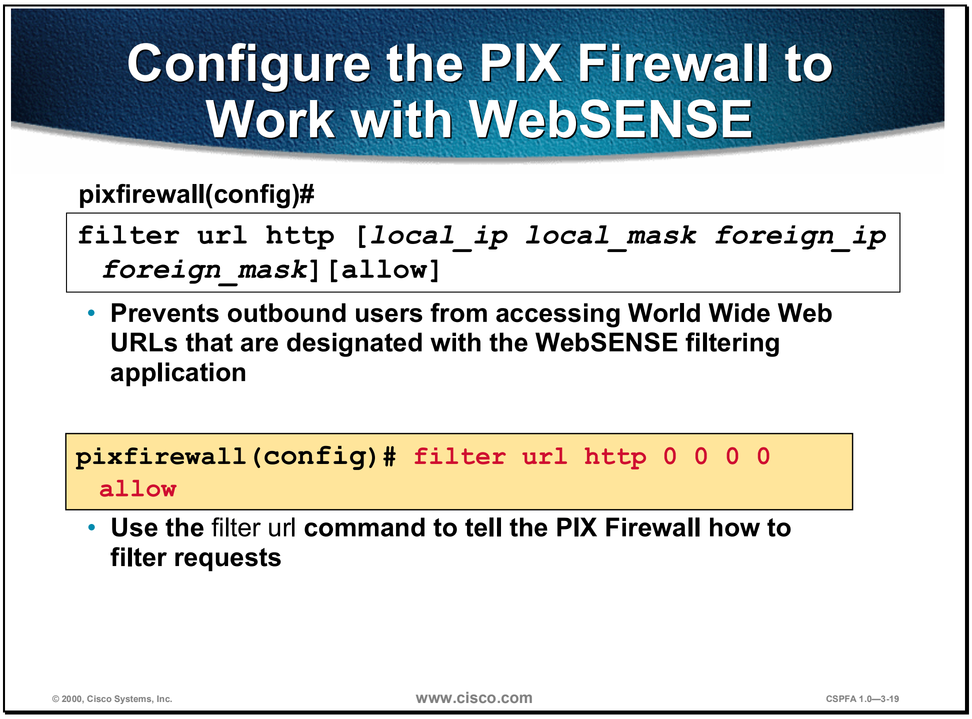
- In this example, the WebSENSE host is on the inside interface at IP address 10.0.0.3. A time value of 10 seconds is specified as the maximum allowed idle time before the PIX Firewall switches to the next WebSENSE server.

Before you can begin URL filtering by configuring WebSENSE or downloading the Master Database for WebSENSE, you must designate the server on which WebSENSE will be running. Use the **url-server** command to designate the server that WebSENSE runs on, then enable the URL filtering service with the **filter url** command.

The syntax for the **url-server** command is as follows:

url-server [(*if_name*)] host *ip_address* [timeout *seconds*]

Argument	Description
if_name	The network interface where the authentication server resides. If not specified, the default is inside.
host *ip_address*	The server that runs the WebSENSE URL filtering application.
timeout *seconds*	The maximum idle time permitted before the PIX Firewall switches to the next server you specified. The default is 5 seconds.



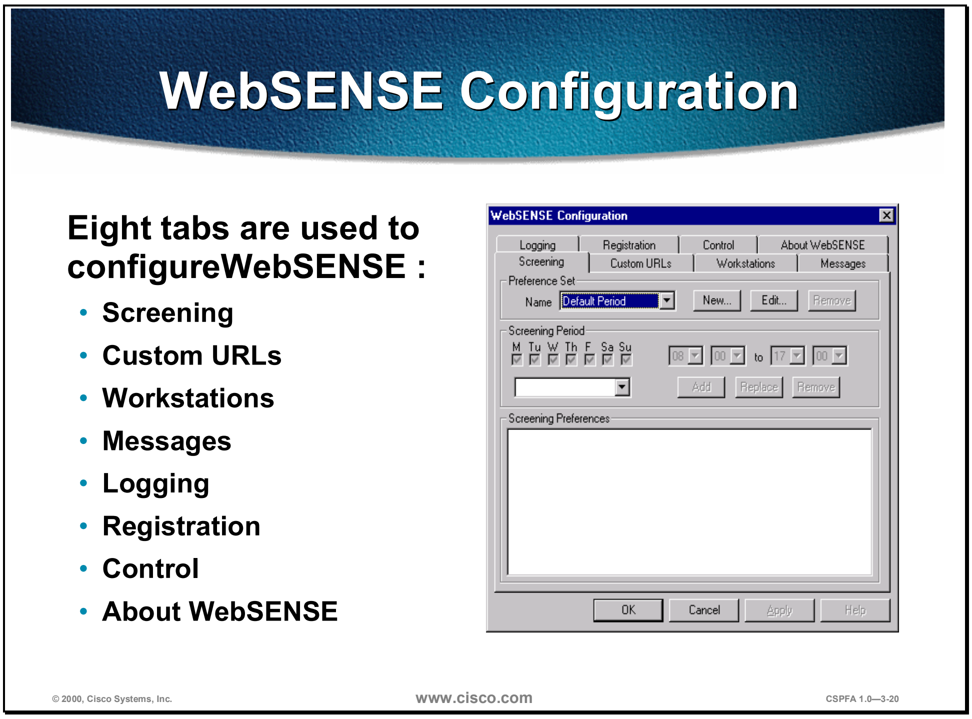
After designating which server uses WebSENSE, use the **filter url** command to tell the PIX Firewall to send URL requests to WebSENSE for filtering.

The example command in the figure above instructs the PIX Firewall to send all URL requests to the WebSENSE server to be filtered. The **allow** option in the filter command is crucial to the use of the PIX Firewall URL filtering feature. If you use the **allow** option and the WebSENSE server goes offline, the PIX Firewall lets all URL requests continue without filtering. If the **allow** option is not specified, all port 80 URL requests are stopped until the server is back online.

The syntax for the **filter url** command is as follows:

filter url http | except ***local_ip local_mask foreign_ip foreign_mask*** **[allow]**

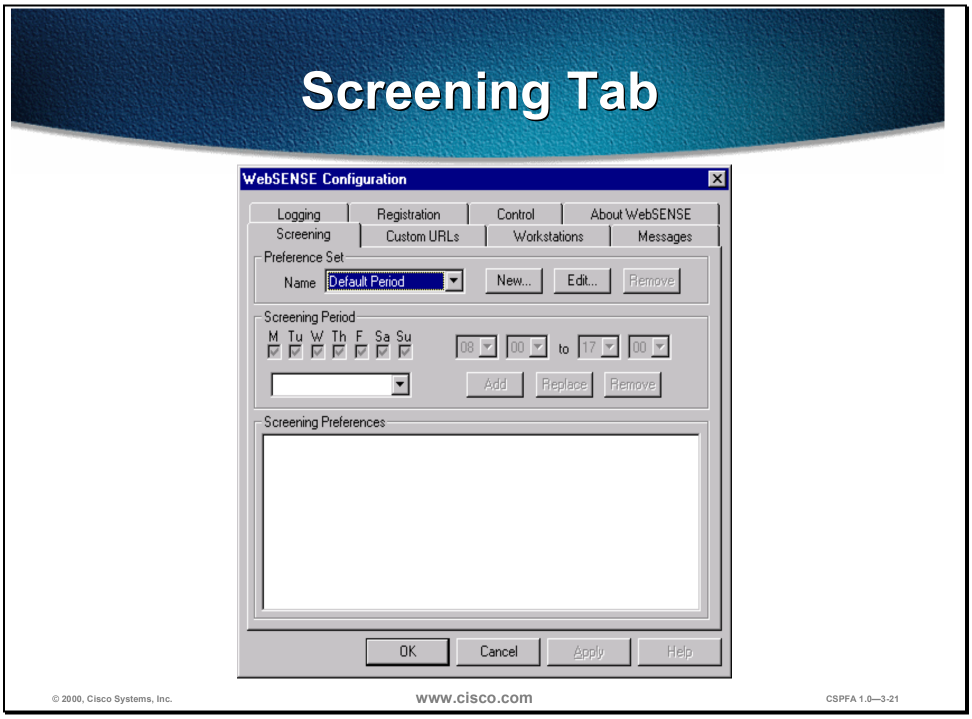
Argument	Description
url	Filter URLs from data moving through the PIX Firewall.
http	Filter HTTP (World Wide Web) URLs.
except	Create an exception to a previous **filter** condition.
local_ip	The IP address of the highest security level interface from which access is sought.
local_mask	Network mask of ***local_ip***.
foreign_ip	The IP address of the interface with the lowest security level to which access is sought.
foreign_mask	Network mask of ***foreign_ip***.
allow	When the WebSENSE server is unavailable, let outbound connections pass through the PIX Firewall without filtering.



When you open the WebSENSE Configuration window, you will see eight configuration tabs:

- Screening—Gives you control over the protocols and categories that WebSENSE blocks and the times when they are blocked.
- Custom URLs—Enables you to override blocking of web sites, and allows you to screen web sites that are not in the WebSENSE Master Database.
- Workstations—Enables you to create different access policies on a workstation-by-workstation basis.
- Messages—Enables you to create a message that displays when a user is trying to access a web page to which access is not permitted.
- Logging—Enables you to keep track of Internet activity passing through the PIX Firewall.
- Registration—Enables you to enter and modify WebSENSE registration information.
- Control—Displays the version of the Master Database is currently running.
- About WebSENSE—Provides the information you need to contact NetPartners for WebSENSE-related sales and support.

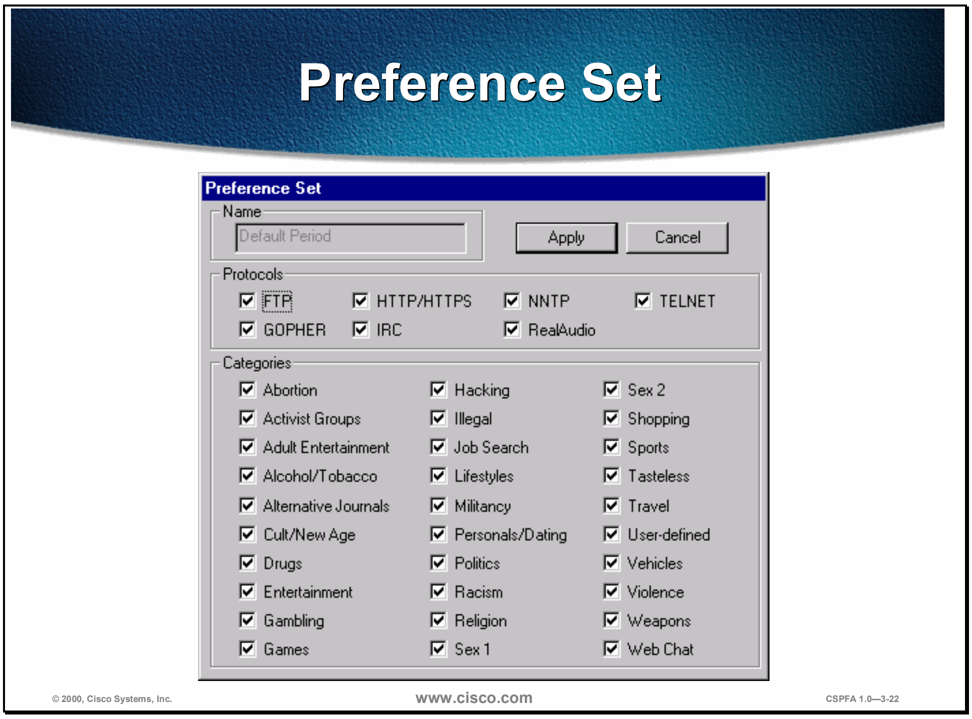
CAUTION *If you want to save any changes that you have made to the WebSENSE configuration using the eight tabs, you MUST stop and then restart the WebSENSE server after your changes are complete.*



Use the Screening tab to create preference sets, which give you control over the protocols and categories that WebSENSE blocks and the times when they are blocked. The following are three of the frames found in the Screening tab:

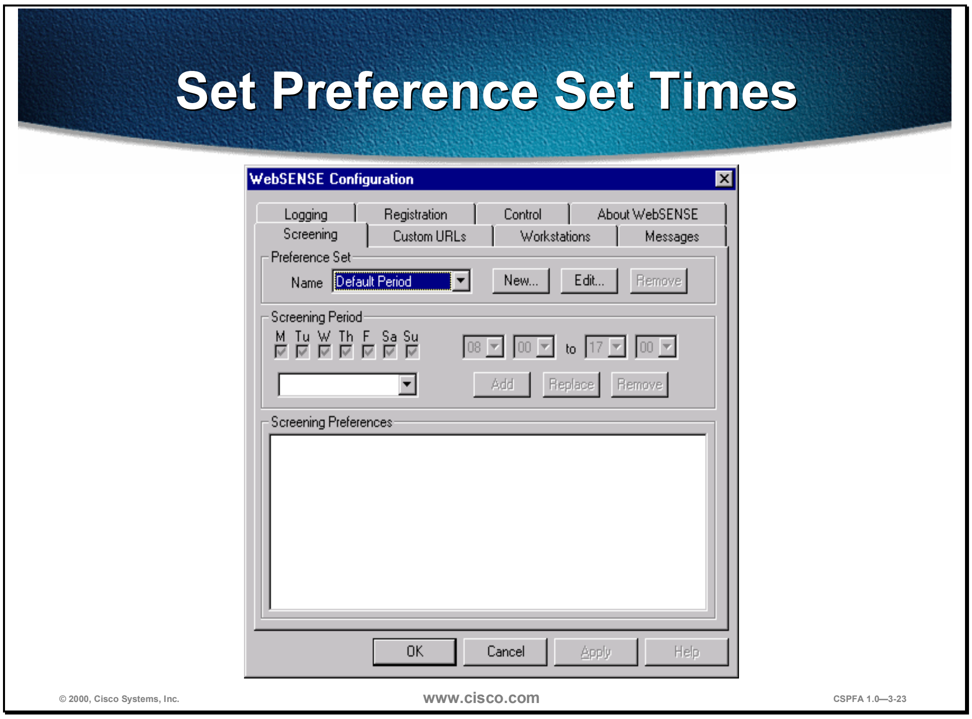
- Preference Set—Controls what protocols and categories are blocked by each preference set.
- Screening Period—Specifies the active periods for the preference sets previously defined.
- Screening Preferences—Lists the screening periods you have established.

Notice the entry *Default Period* in the Name field within the Preference Set frame. This appears in the Name field by default, and is a predefined preference set. It is in effect when no other preference set is active.



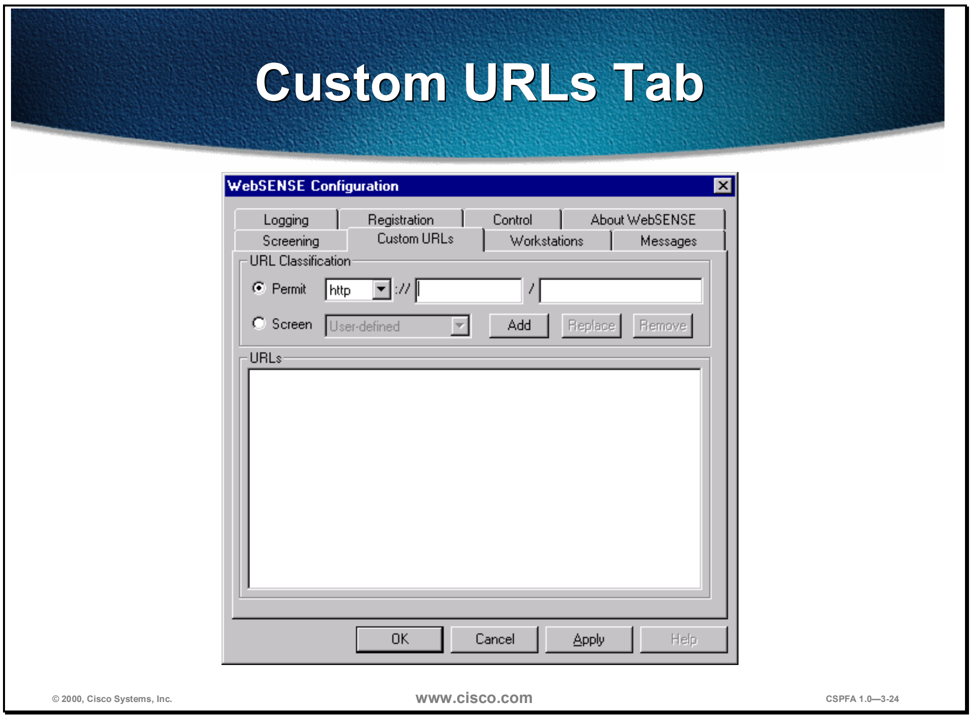
After you choose whether you want to create a new preference set or edit an existing preference set by clicking either **New** or **Edit** from the Preference Set frame in the Screening tab, the Preference Set window opens. From this window you define your new preference set or edit your existing preference set. Specifically, you can choose what protocols and categories to block by selecting or deselecting the checkboxes within both the Protocols and Categories frames.

If you choose to edit the Default Period preference set, initially all protocols and categories are blocked (all checkboxes are selected). You can modify the protocols and categories that the Default Period blocks, but you cannot set its active period.



After you have defined a preference set, you can create one or more screening periods for it. Each screening period identifies the time of day and days of the week that the preference set is active. Choose the days and times when the preference set will be active from the Screening Period frame in the Screening tab. When a preference set is active, WebSENSE blocks the categories and protocols that have been selected in the Preference Set window.

Note Remember that the screening dates and times displayed in the Screening Period frame are never applicable to the Default Period, which is automatically active when no other preference set is active.

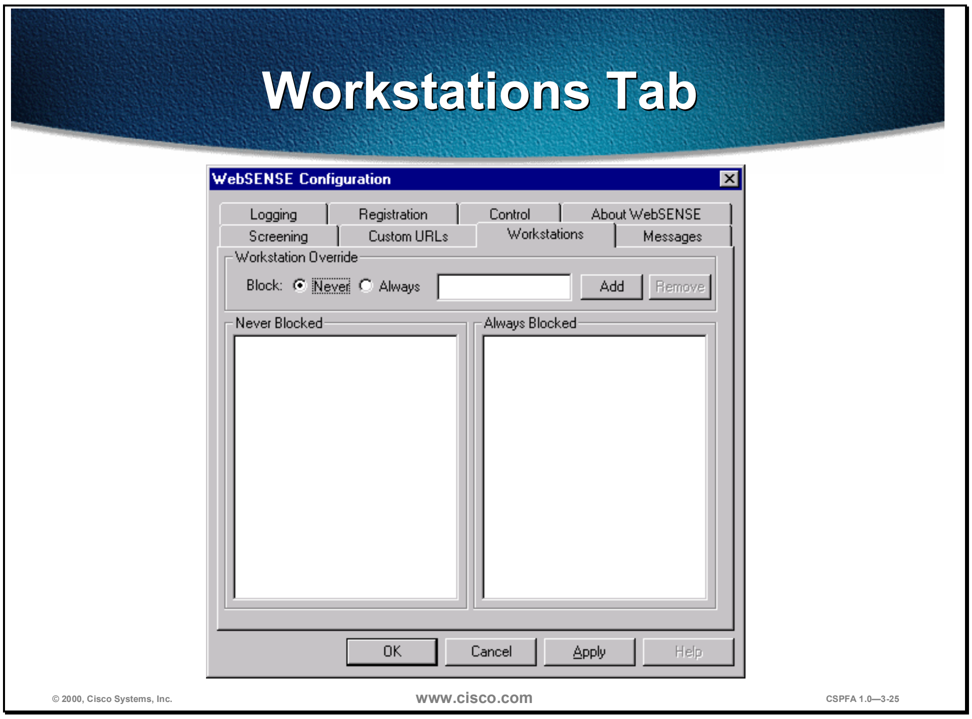


When using the Master Database—the list of URLs that should be blocked as determined and updated daily by the WebSENSE corporate office—you may need to permit one of the blocked URLs. If you need to permit URLs that are normally blocked by the parameters of the Master Database, you can add the URLs to a special “permit list.” URLs in this list will never be blocked by WebSENSE. When accessing a URL that has been added to the permit list, WebSENSE logs the access as normal.

To add a URL to the permit list, click **Permit** in the Custom URLs tab. Choose which protocol to use from the drop-down list next to Permit, then enter the URL in the field that follows. Click the **Add** button. The URL then appears in the URLs window with the designation “Permit.”

Sometimes certain URLs are not blocked by either the WebSENSE configuration or the Master Database. If you need a specific URL blocked, add the URL to a special “screen list.” After these URLs have been added, when users attempt to access the URLs specified on the screen list, they will be blocked and the event logged by WebSENSE.

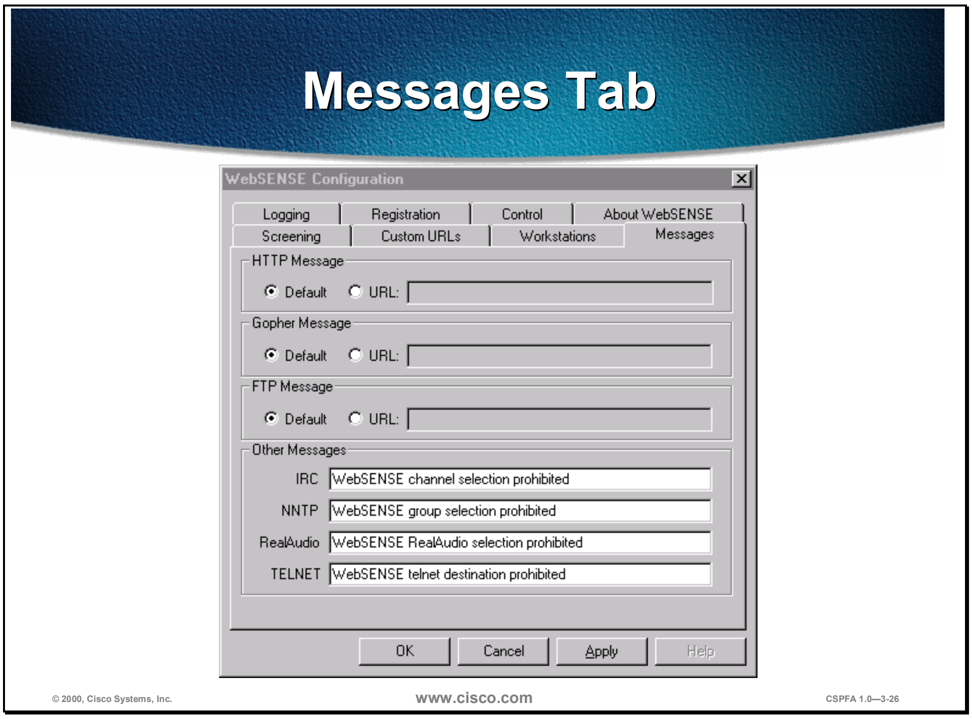
To screen a URL, click **Screen** in the Custom URLs tab. Choose the category under which to track the site from the drop-down list next to Screen. (There is a special category called *User Defined*, which can be used as a general purpose category to hold miscellaneous URLs.) Choose which protocol to use from the drop-down list next to Permit, then enter the URL in the field that follows. Click the **Add** button. The URL then appears in the URLs window with the designation “Screen.”



The Workstations tab enables you to determine a workstation's Internet access capabilities. You can give a workstation unrestricted access to the Internet, or block a workstation from any access to the Internet.

When workstations with unrestricted Internet access try to access a URL that is normally blocked by WebSENSE, their access is logged by WebSENSE as an *observed access*. To give a workstation unrestricted access, select **Never** in the Workstation Override frame, enter the workstation's IP address in the field that follows, and click **Add**.

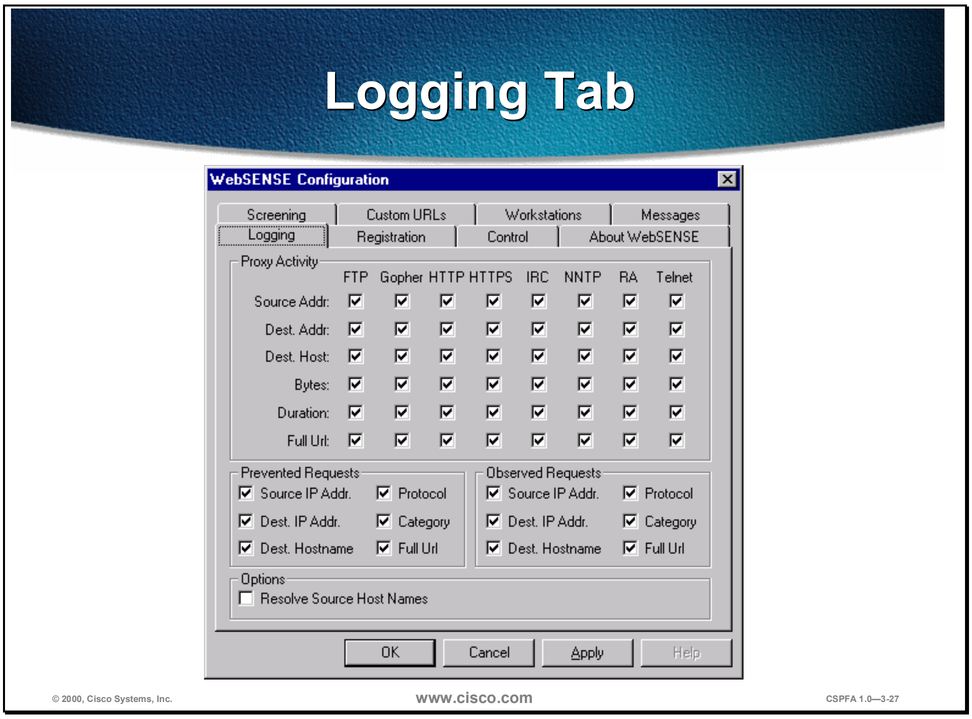
Workstations that have been restricted from accessing the Internet completely will not have HTTP access to the Internet. To restrict Internet access for a workstation, select **Always** in the Workstation Override frame, then enter that workstation's IP address in the field that follows, and click the **Add** button. The IP address will appear in the Always Blocked frame.



By default, when you attempt to access a web site that WebSENSE is blocking, you receive the message "Access to the desired web page is not permitted". Instead of using the default message, you can send the user who is trying to access the blocked URL to a different web page. For example, if a user tries to access a blocked web page, they are sent to a web site that explains that access to the web site is not permitted, including any other information you wish.

To send the user who is trying to access the blocked URL to a different web page, select **URL** in the HTTP Message frame and then enter the URL in the field that follows..

The rest of the fields in the Messages tab are not usable because the PIX Firewall does not support the filtering of other protocols at this time.

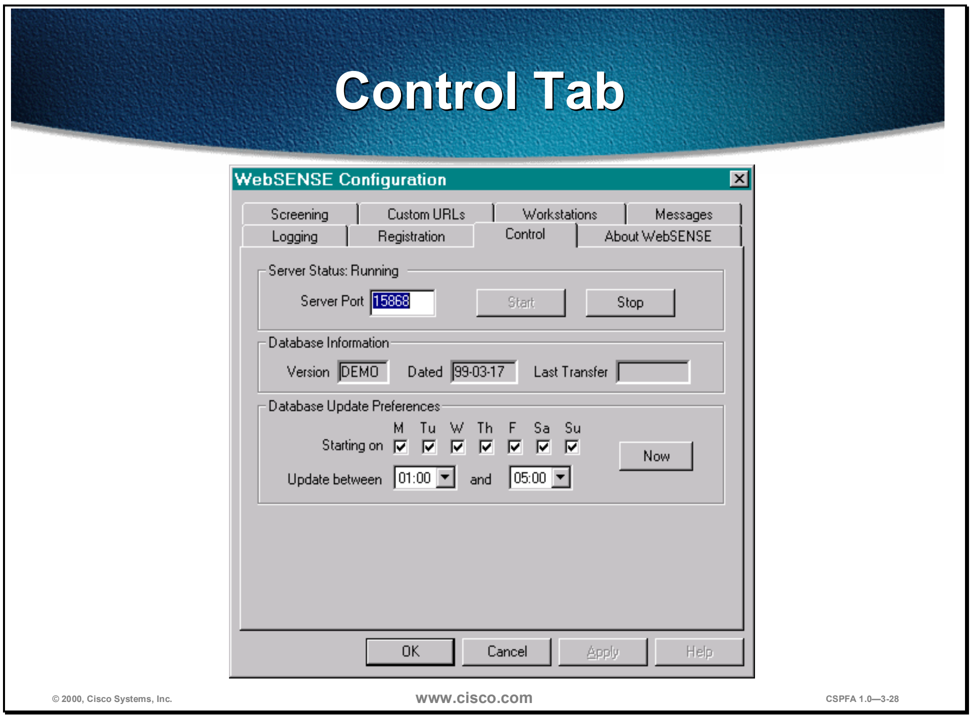


WebSENSE enables you to log URL requests. Specifically, use the Logging tab to specify whether you want to log information for prevented requests (requests for blocked URLs), observed requests (requests for URLs that are normally blocked, but have been made accessible), or both. You can also choose exactly which elements of the Internet request you want to log.

There are four frames within the Logging tab:

- Proxy Activity—Enables you to log attempted access-restricted proxies by source address, destination address, destination host, and bytes.
- Prevented Requests—Enables you to log attempted access to sites blocked by WebSENSE.
- Observed Request—Enables you to log access to sites that are contained in the WebSENSE database, but were not set up for blocking at the time of the request.
- Options—Enables you to change how logging is performed. If the Resolve Source Host Names checkbox is selected, WebSENSE performs a reverse IP address lookup and logs the workstation's domain name as well as the IP address.

WebSENSE writes log entries to a text file called WebSENSEstats.log, located in the WebSENSE directory. The WebSENSE Reporter imports these logs and produces reports in graphical or tabular formats.



The Control tab lists the version of the Master Database that is currently running and allows you to choose the day and time when WebSENSE will download the latest Master Database version. By default, WebSENSE contacts the Master Database, at the WebSENSE corporate headquarters, every day via the Internet to download the latest version.

Note The WebSENSE preference sets that you configure are never deleted when the new version of the Master Database is downloaded.

There are three frames within the Control tab:

- Server Status Running—Contains the server port information for WebSENSE corporate headquarters, and enables you to start and stop the server.
- Database Information—Displays the version of the Master Database that is currently running.
- Database Update Preference—Enables you to set the time period each day during which WebSENSE will contact the Master Database for download.

WebSENSE chooses a random time during the time interval specified in the Update between boxes within the Database Update Preferences frame to attempt the download of the latest Master Database. If WebSENSE is unable to contact the Master Database at the scheduled time, it will retry every 10 minutes during the specified times until a new copy of the Master Database is successfully downloaded. If a Master Database download is unsuccessful, the Dated field is not updated.

You can set your own date and time for WebSENSE to download the latest Master Database by setting the parameters in the Database Update Preferences frame. You can also download the Master Database at any time by clicking the **Now** button in the Database Update Preferences frame.

HTTP Fixup Configuration

pixfirewall (config)#

```
fixup protocol http [port[-port]
```

- Defines ports for HTTP connections (default = 80)
  - Logs all URLs accessed in HTTP traffic (when syslog is enabled)
  - Enables URL-based filtering (WebSENSE, Java, ActiveX)
- If disabled,
  - Stops URL logging
  - URL-based filtering is disallowed

```
pixfirewall(config)# fixup protocol http 8080
pixfirewall(config)# fixup protocol http 8180-8200
pixfirewall(config)# no fixup protocol http 80
```

 www.cisco.com CSPFA 1.0—3-29

The HTTP fixup protocol is enabled by default. The default port for HTTP connections is port 80. When the HTTP fixup protocol is enabled, it logs all URLs accessed in HTTP traffic (when syslog is enabled). It also enables URL-based filtering.

If the HTTP fixup protocol is disabled, URL logging and URL-based filtering stops.

The syntax for the **fixup protocol http** command is as follows:

fixup protocol http [*port*[-*port*]

Argument	Description
protocol	Specifies the protocol to fix up.
port	Specifies the port number or range for the application protocol.

Lab Exercise: Configure WebSENSE

Complete the following lab exercise to practice what you learned in this chapter.

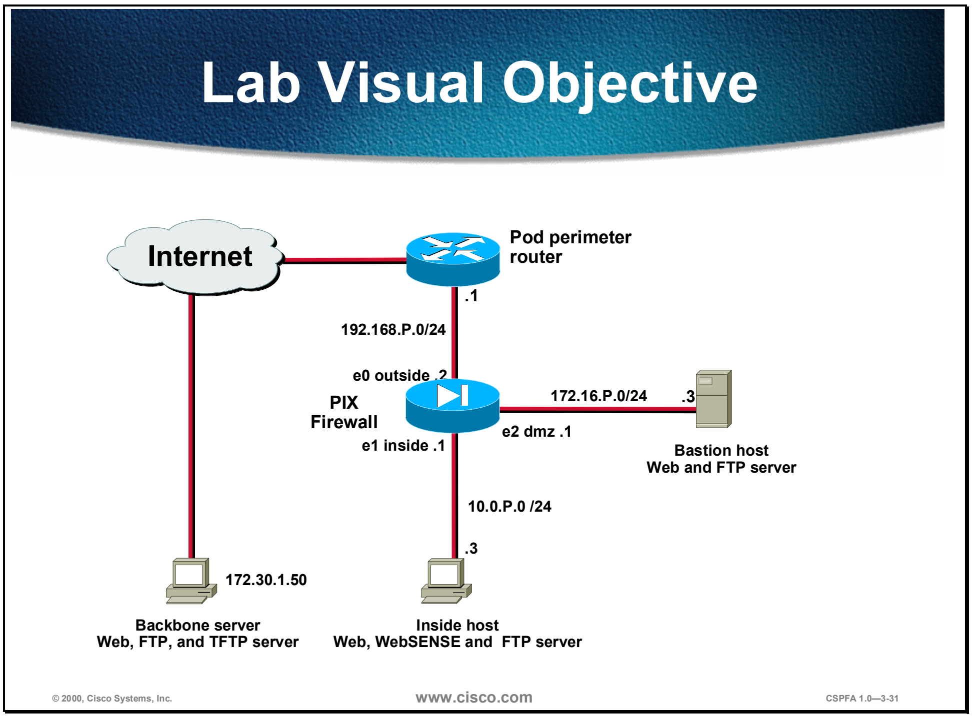
Objectives

In this lab exercise you will complete the following tasks:

- Configure the ACL.
- Filter malicious active code.
- Configure the PIX Firewall to work with WebSENSE.
- Install WebSENSE on a Windows NT Server.
- Configure WebSENSE to block a web site.

Visual Objective

The following figure displays the configuration you will complete in this lab exercise.



Task 1: Configure the ACL

Perform the following lab steps to configure the access list to stop web traffic, but to allow other IP traffic through the PIX Firewall:

Step 1 Test connections to 172.30.1.50 by using FTP and HTTP connections.

Step 2 Enter the **access-list** command to create an ACL that will deny internal network Internet access:

```
pixfirewall(config)# access-list 101 deny tcp any any eq www
```

Step 3 Enter the **access-group** command to create an access group that will bind the ACL to an interface:

```
pixfirewall(config)# access-group 101 in interface inside
```

Step 4 Test connections to 172.30.1.50 by using FTP and HTTP connections.

Step 5 Remove the **access-group** command:

```
pixfirewall(config)# no access-group 101 in interface inside
```

Step 6 Add an additional command to the ACL:

```
pixfirewall(config)# access-list 101 permit tcp any any eq ftp
```

Step 7 Bind the ACL to an interface by creating an access group:

```
pixfirewall(config)# access-group 101 in interface inside
```

Step 8 Test connections to 172.30.1.50 by using FTP and HTTP connections.

Step 9 Remove *access-list 101* from the PIX Firewall:

```
pixfirewall(config)# clear access-list
```

Step 10 Show the access list:

```
pixfirewall(config)# show access-list
```

Step 11 Show the access-group:

```
pixfirewall(config)# show access-group
```

Task 2: Filter Malicious Active Code

Perform the following lab steps to configure ActiveX and filter Java. You will not be able to test this task:

Step 1 Enter the **filter activex** command to block ActiveX from any local host and for connections to any foreign host on port 80:

```
pixfirewall(config)# filter activex 80 0 0 0 0
```

Step 2 Enter the **filter java** command to block Java applets:

```
pixfirewall(config)# filter java 80 0 0 0 0
```

Step 3 Use the following command to show you the filters:

```
pixfirewall(config)# show filter
filter activex 80 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
filter java 80 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
```

Task 3: Configure the PIX Firewall to Work with WebSENSE

Perform the following steps to configure the PIX Firewall to work with the WebSENSE server:

Step 1 Enter the **configure terminal** command to enter config mode:

```
pixfirewall(config)# config terminal
```

Step 2 Enter the **url-server** command to designate the WebSENSE server:

```
pixfirewall(config)# url-server (inside) host 10.0.P.3
```

(where *P* = pod number)

Step 3 Show the desinated url-server by entering the following command:

```
pixfirewall(config)# show url-server
url-server (inside) host 10.0.1.3 timeout 5
```

Step 4 Enter the **filter url** http command to prevent outbound users from accessing World Wide Web URLs that are designated with the WebSENSE filtering application:

```
pixfirewall(config)# filter url http 0 0 0 0 allow
```

Step 5 Display the **filter url http** command by using the following command

```
pixfirewall(config)# show filter url
filter url http 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 allow
```

Task 4: Install WebSENSE on an NT Server

Perform the following steps to install the WebSENSE software onto the NT server. Accept all of the default settings for the software installation:

Step 1 Double-click the **WebSENSE** folder located on the desktop of the computer.

Step 2 Double-click **wspix303.exe**. This begins the installation process for WebSENSE.

Step 3 When the WebSENSE for Cisco PIX window opens, click **Next** to continue.

Step 4 Click **Yes** to accept the terms of the WebSENSE software license agreement.

Step 5 Click **Next** to accept the default destination folder.

Step 6 Set the password for the server:

```
Password: cisco
Password (again): cisco
```

Step 7 Click **Next** to continue.

Step 8 Click **Next** to accept the default port of *18072*.

Step 9 Click **Next** to accept the default components to install.

Step 10 Click **Finish** to complete the installation.

Task 5: Configure WebSENSE to Block a Web Site by URL

Complete the following steps to configure WebSENSE to block a web site by URL:

Step 1 Test the HTTP connection to 172.30.1.50 in your web browser.

Step 2 After making sure you can reach this web page, close your browser.

Step 3 Open the WebSENSE Control Panel by choosing **Start>Programs>WebSENSE for Cisco Pix>WebSENSE Control Panel**.

Step 4 In the WebSENSE Control Panel, select the **Custom URLs** tab.

Step 5 In the URL Classification field, select **Screen** and in the field following the http drop-down menu, enter **172.30.1.50** to block external web sites.

Step 6 Click **Add**.

Step 7 Click **Apply**.

Step 8 Click **OK**. The Restart Server window opens.

Step 9 Click **OK**.

Step 10 Test the HTTP connection to 172.30.1.50. You should not be able to open the web page.

Step 11 Close your browser.

Step 12 Open the WebSENSE Control Panel by choosing **Start>Programs>WebSENSE for Cisco Pix>WebSENSE Control Panel**.

Step 13 In the WebSENSE Control Panel, select the **Custom URLs** tab.

Step 14 In the URLs field, select the rule you just created and click **Remove**.

Step 15 Click **Apply**.

Step 16 Click **OK**. The Restart Server window opens.

Step 17 Click **OK**.

Step 18 Test the HTTP connection to 172.30.1.50 You should be able to open the web page.

Step 19 After making sure you can reach this web page, close your browser.

Task 6: Configure WebSENSE to Block a Web Site by Workstation

Complete the following steps to configure WebSENSE to block a workstation by IP address from accessing theInternet.

Step 1 Open the WebSENSE Control Panel by choosing **Start>Programs>WebSENSE for Cisco Pix>WebSENSE Control Panel**.

Step 2 From the Workstations tab in the Workstation Override frame, select **Always** and enter your workstation IP address **10.0.P.3** in the field that follows.

(where*P* = pod number)

Step 3 Click **Add**.

Step 4 Click **Apply**.

Step 5 Click **OK**. The Restart Server window opens.

Step 6 Click **OK**.

Step 7 Test the HTTP connection to 172.30.1.50. You should not be able to open the web page.

Step 8 Close your browser.

Step 9 Open the WebSENSE Control Panel by choosing **Start>Programs>WebSENSE for Cisco Pix>WebSENSE Control Panel**.

Step 10 From the Workstations tab in the Always Blocked frame, select the workstation IP address you just created.

Step 11 Click **Remove**.

Step 12 Click **Apply**.

Step 13 Click **OK**. The Restart Server window opens.

Step 14 Click **OK**.

Step 15 Open the WebSENSE Control Panel by choosing **Start>Programs>WebSENSE for Cisco Pix>WebSENSE Control Panel**.

Step 16 Click the **Control** tab and then the **Stop** button to stop the WebSENSE server.

Step 17 Click **OK**.

Task 7: Reset the PIX Firewall and the WebSENSE Server

Perform the following to reset the PIX Firewall and the WebSENSE server:

Step 1 Remove the url-server:

```
pixfirewall(config)# no url-server (inside) host 10.0.P.3
```

(where P = your pod number)

Step 2 Remove the **filter url** command:

```
pixfirewall (conifg)# no filter url http 0 0 0 0 allow
```

Step 3 Open the WebSENSE Control Panel by choosing Start>Programs>WebSENSE for Cisco Pix>Uninstall WebSENSE for Cisco PIX

Step 4 You are prompted, **Are you sure you want to completely remove "WebSENSE for Cisco PIX" and all of its components?** Click **YES**.

Step 5 You are prompted whether you want to **Remove Shared File**. Click **Yes To All**.

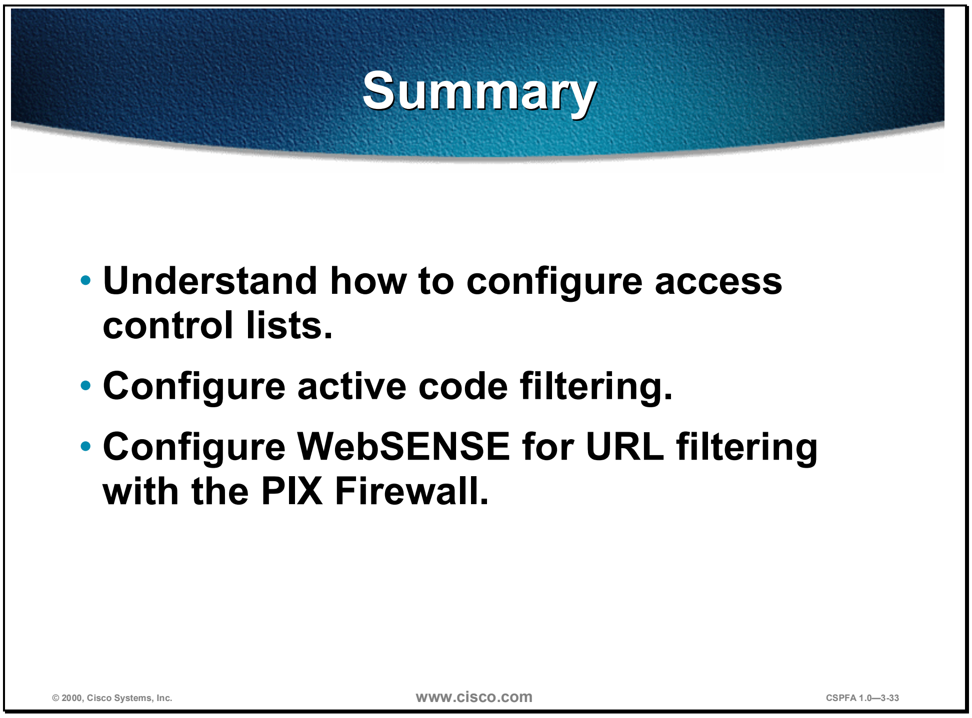
Step 6 You are then asked to verify that you want to remove the shared file. Click **Yes**.

Step 7 Click **OK**.

Step 8 You are prompted to restart you computer. Reboot your computer.

Summary

This section summarizes the tasks you learned to complete in this chapter.



AAA Configuration on the Cisco Secure PIX Firewall

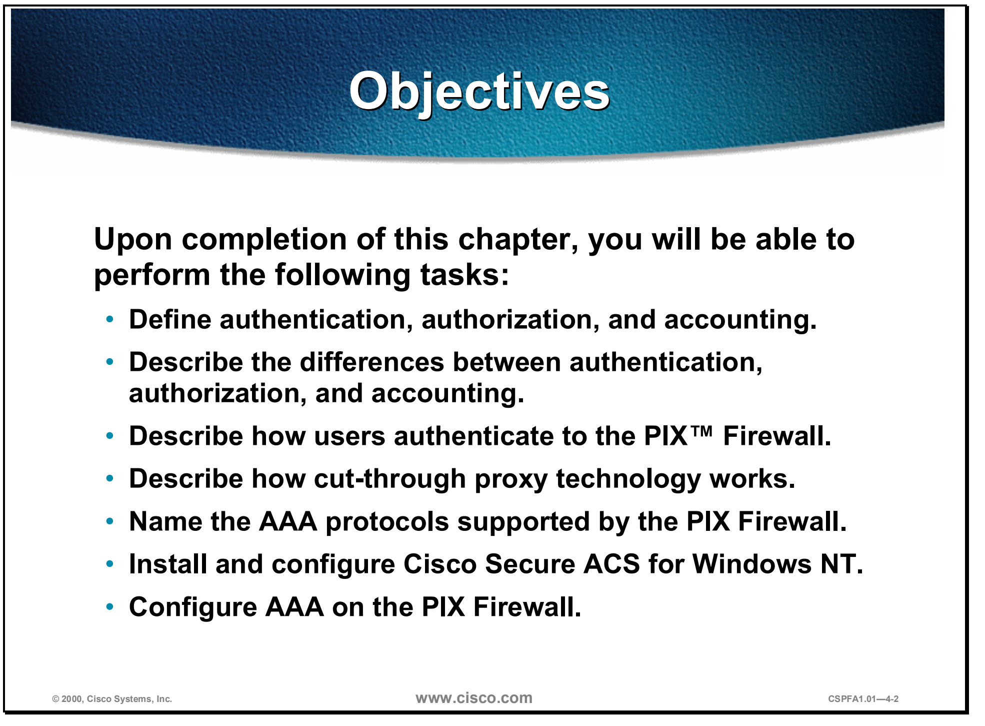
Overview

This chapter includes the following topics:

- Objectives
- Introduction
- Installation of Cisco Secure ACS for Windows NT
- Authentication configuration
- Authorization configuration
- Accounting configuration
- Troubleshooting the AAA configuration
- Lab exercise
- Summary

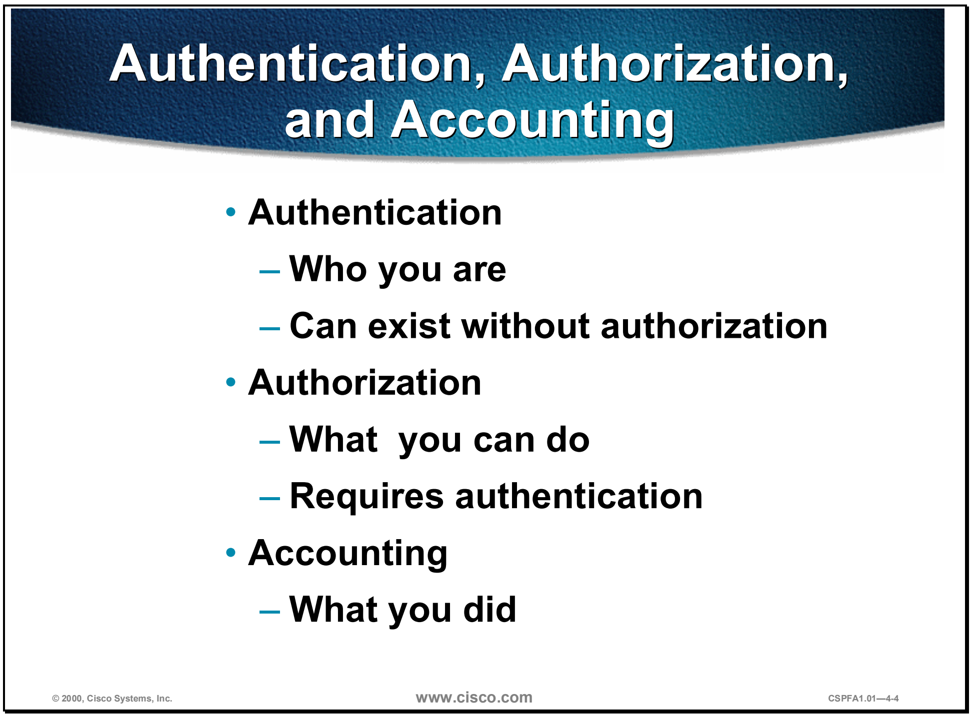
Objectives

This section lists the chapter's objectives.



Introduction

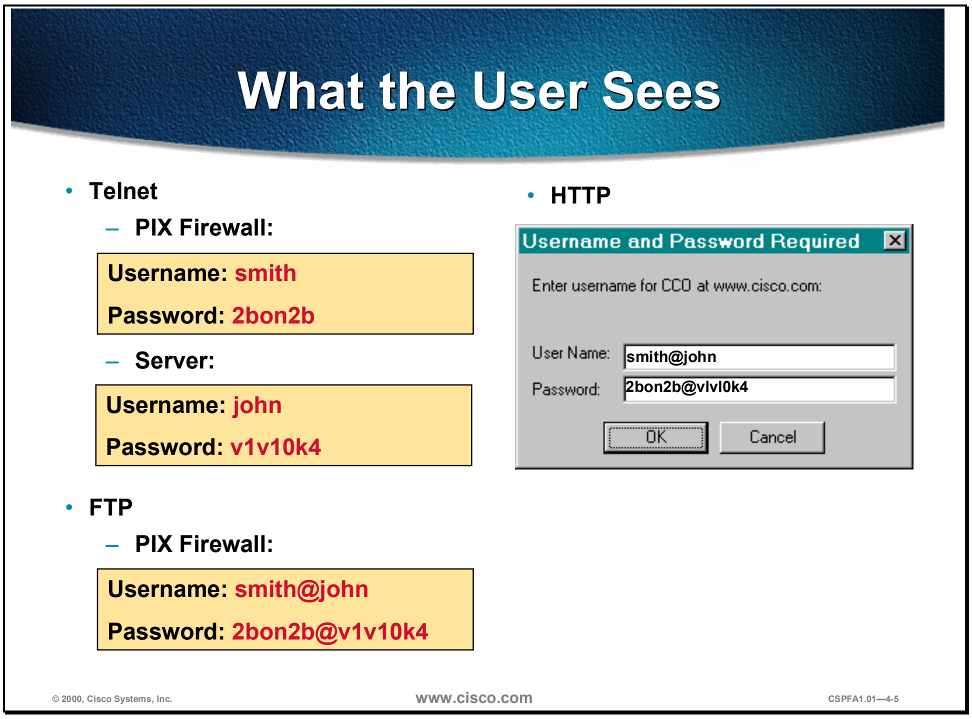
This section introduces the authentication, authorization, and accounting concepts and how the PIX™ Firewall supports them.



Authentication, Authorization, and Accounting (AAA) is used to tell the PIX Firewall who the user is, what the user can do, and what the user did. Authentication is valid without authorization. Authorization is never valid without authentication.

Suppose you have 100 users inside and you want only six of these users to perform FTP, Telnet, or HTTP outside the network. Tell the PIX Firewall to authenticate outbound traffic and give all 6 users identifications on the TACACS+ or RADIUS AAA server. With simple authentication, these six users are authenticated with a username and password, and then permitted outside the network. The other 94 users cannot go outside the network. The PIX Firewall prompts users for their username and password, then passes their username and password to the TACACS+ or RADIUS AAA server. Depending on the response, the PIX Firewall opens or denies the connection.

Suppose one of these users, “baduser,” is not to be trusted. You want to allow “baduser” to perform FTP, but not HTTP or Telnet to the outside. This means you must add authorization, that is, authorize what users can do in addition to authenticating who they are. This is only valid with TACACS+. When you add authorization to the PIX Firewall, it first sends the untrusted user a username and password to the AAA server, then sends an authorization request telling the AAA server what command “baduser” is trying to do. With the server set up properly, “baduser” is allowed to perform FTP but is not allowed to perform HTTP or Telnet.



You can authenticate with the PIX Firewall in one of three ways:

- Telnet—You get a prompt generated by the PIX Firewall. You have up to four chances to log in. If the username or password fail after the fourth attempt, the PIX Firewall drops the connection. If authentication and authorization are successful, you are prompted for a user name and password by the destination server.
- FTP—You get a prompt from the FTP program. If you enter an incorrect password, the connection is dropped immediately. If the username or password on the authentication database differs from the username or password on the remote host to which you are accessing via FTP, enter the username and password in the following formats:
  - aaa_username@remote_username
  - aaa_password@remote_password

  The PIX Firewall sends the aaa_username and aaa_password to the AAA server, and if authentication and authorization are successful, the remote_username and remote_password are passed to the destination FTP server beyond.

Note Some FTP graphical user interfaces (GUIs) do not display challenge values.

- HTTP—You see a pop-up window generated by the web browser. If you enter an incorrect password, you are prompted again. If the username or password on the authentication database differs from the username or password on the remote host to which you are using HTTP to access, enter the username and password in the following formats:
  - aaa_username@remote_username
  - aaa_password@remote_password

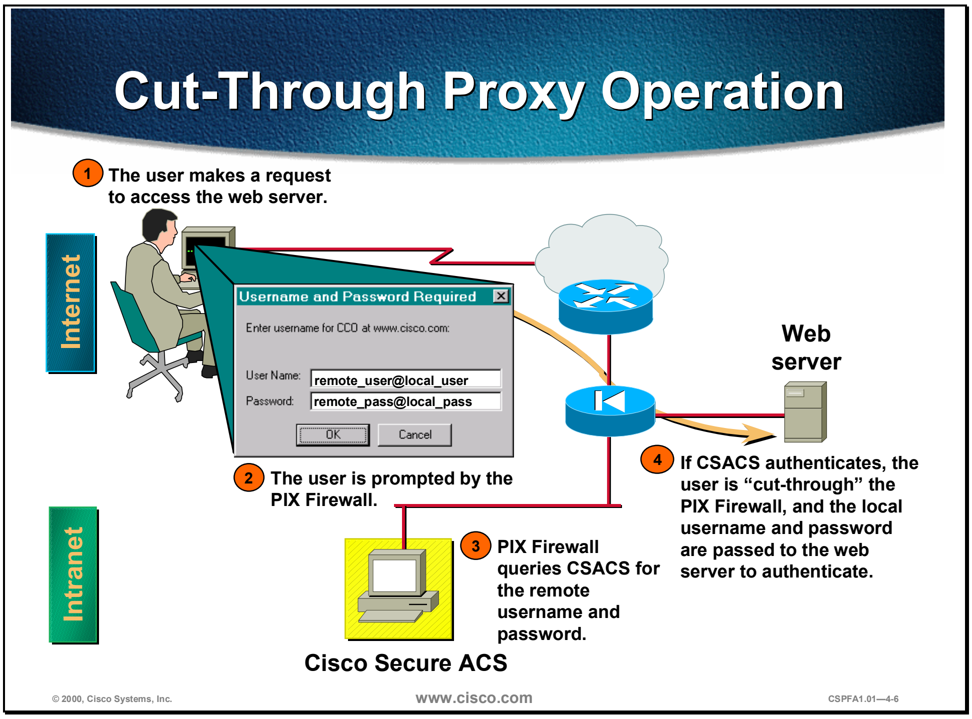
The PIX Firewall sends the aaa_username and aaa_password to the AAA server, and if authentication and authorization are successful, the remote_username and remote_password are passed to the destination HTTP server.

Keep in mind that browsers cache usernames and passwords. If you believe that the PIX Firewall should be timing out an HTTP connection but it is not, re-authentication may actually be taking place with the web browser sending the cached username and password back to the PIX Firewall. The SYSLOG service will show this phenomenon. If Telnet and FTP seem to work normally, but HTTP connections do not, this is usually why.

The PIX Firewall supports authentication usernames up to 127 characters and passwords of up to 63 characters. A password or username may not contain an at (@) character as part of the password or username string.

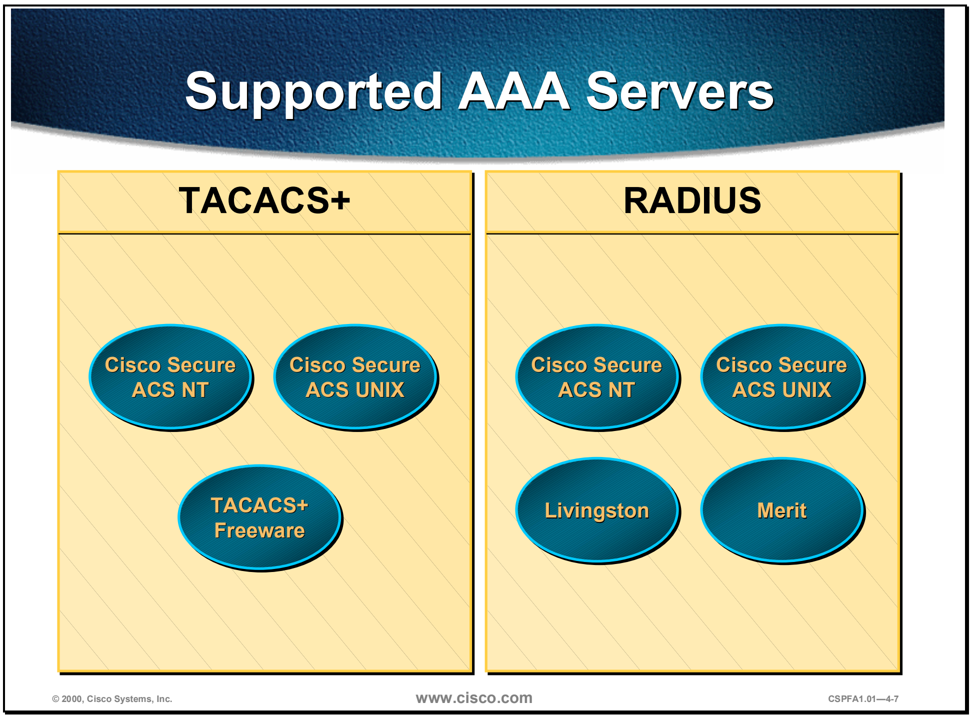
Note If PIX Firewalls are in tandem, Telnet authentication works in the same way as a single PIX Firewall, but FTP and HTTP authentication have additional complexity because you have to enter each password and username with an additional at "@" character and password or username for each in-tandem PIX Firewall.

Note Once authenticated with HTTP, a user never has to reauthenticate no matter how low the PIX Firewall uauth timeout is set. This is because the browser caches the "Authorization: Basic=Uuhjksdkfhk==" string in every subsequent connection to that particular site. This can only be cleared when the user exits all instances of Netscape Navigator or Internet Explorer and restarts. Flushing the cache is of no use.



The PIX Firewall gains dramatic performance advantages because of cut-through proxy, a patent-pending method of transparently verifying the identity of users at the firewall and permitting or denying access to any TCP- or UDP-based application. This method eliminates the price and performance impact that UNIX system-based firewalls impose in similar configurations, and leverages the authentication and authorization services of the Cisco Secure Asynchronous Communications Server (ACS).

The PIX Firewall's cut-through proxy challenges a user initially at the application layer, then authenticates against standard TACACS or RADIUS+ databases. After the policy is checked, the PIX Firewall shifts the session flow, and all traffic flows directly and quickly between the server and the client while maintaining session state information.

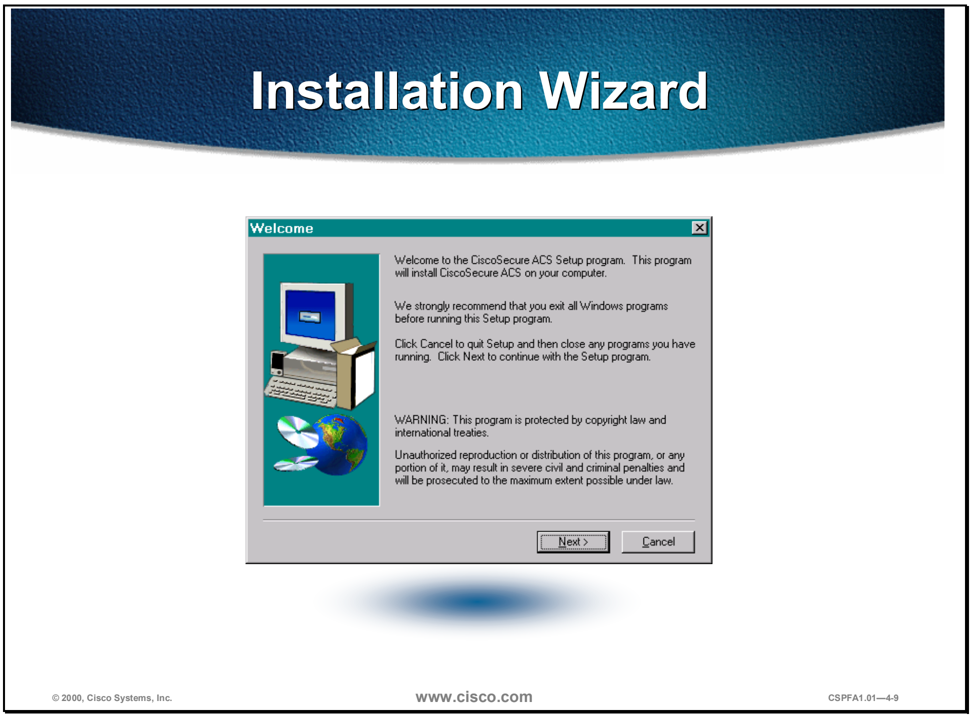


The PIX Firewall supports the following AAA protocols and servers:

- Terminal Access Controller Access Control System Plus (TACACS+)
  - Cisco Secure Asynchronous Communications Server (CSACS) for Windows NT (CSACS-NT)
  - Cisco Secure ACS for UNIX (CSACS-UNIX)
  - TACACS+ Freeware
- Remote Authentication Dial-In User Service (RADIUS)
  - Cisco Secure ACS for Windows NT (CSACS-NT)
  - Cisco Secure ACS for UNIX (CSACS-UNIX)
  - Livingston
  - Merit

Installation of Cisco Secure ACS for Windows NT

This section explains how to install the Cisco Secure ACS for Windows NT.



Note Close all Windows programs before you run Setup.

To start installation of Cisco Secure ACS for Windows NT, complete the following steps:

Step 1 Log in as the local system administrator to the machine on which you are installing Cisco Secure ACS.

Step 2 Insert the Cisco Secure ACS CD-ROM into your CD-ROM drive. The Installation window opens.

Step 3 Click **Install**. The Software License Agreement window opens.

Step 4 Read the Software License Agreement. Click **Accept** to agree to the licensing terms and conditions. The Welcome window opens.

Step 5 Click **Next** The Before You Begin window opens.

Step 6 Verify that each condition is met, and then click the checkbox for each item. Click **Next**

Step 7 Click **Next** (Click **Explain** for more information on the listed items. If any condition is not met, click **Cancel** to exit Setup.)

Step 8 If all conditions are met, click **Next**to continue.

Note If this is a new installation, skip to Step 11.

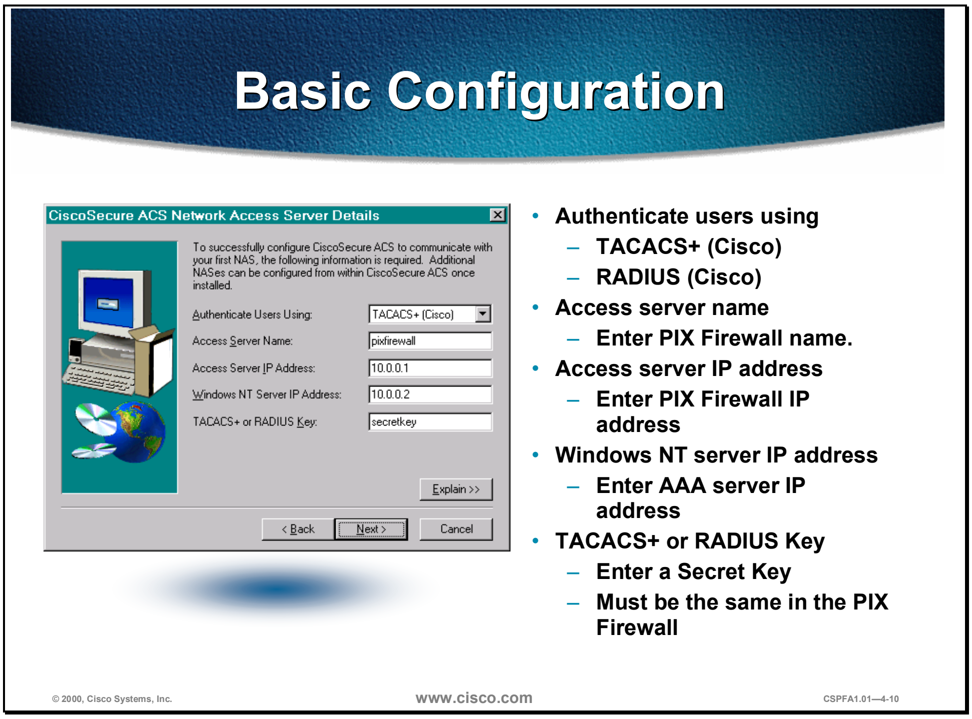
Step 9 (Optional.) If Cisco Secure ACS is already installed, the Previous Installation window opens. You are prompted to remove the previous version and save the existing database information. To keep the existing data, click **Yes, keep existing database** and click **Next**. To use a new database, deselect the checkbox and click **Next** If you checked the c heckbox, Setup backs up the existing configuration. Setup removes the old files. When the files are removed, click **OK**.

Step 10 If Setup finds an existing configuration, you are prompted whether you want to import the configuration. To keep the existing configuration, click **Yes, import configuration** and click **Next** To use a new configuration, deselect the checkbox and click **Next**

Step 11 The Choose Destination Location window opens. To install the software in the default directory, click **Next** To use a different directory, click **Browse** and enter the directory to use. If the directory does not exist, you are prompted to create one. Click **Yes**. The Authentication Database Configuration window opens.

Step 12 Click the option button for the authentication databases to be used by Cisco Secure. Check the Cisco Secure ACS Database only option (the default). Also check the Windows NT User Database option. If you select the first option, Cisco Secure ACS will use only the Cisco Secure ACS database for authentication; if you select the second option, Cisco Secure ACS will check both databases.

Step 13 (Optional.) To limit dial-in access to only those users you specified in the Windows NT User Manager, click the **Yes, reference "Grant dialin permission to user"** setting. Click **Next** The Network Access Serv er Details window opens.



Step 14 Complete the following information:

- Authenticate Users Using—Type of security protocol to be used. TACACS+ (Cisco) is the default.
- Access Server Name—Name of the network access server (NAS) that will be using the Cisco Secure ACS services.
- Access Server IP Address—IP address of the NAS that will be using the Cisco Secure ACS services.
- Windows NT Server IP Address—IP address of this Windows NT server.
- TACACS+ or RADIUS Key—Shared secret of the NAS and Cisco Secure ACS. These passwords must be identical to ensure proper function and communication between the NAS and Cisco Secure ACS. Shared secrets are case sensitive. Setup installs the Cisco Secure ACS files and updates the Registry. Click **Next**

Step 15 The Interface Configuration window opens. The Interface Configuration options are disabled by default. Click the checkbox to enable any or all of the options listed. Click **Next**.

Note Configuration options for these items are displayed in the Cisco Secure ACS interface only if they are enabled. You can disable or enable any or all of these and additional options after installation in the Interface Configuration: Advanced Options window.

Step 16 The Active Service Monitoring window opens. To enable the Cisco Secure ACS monitoring service, CSMon, check the **Enable Log-in Monitoring** checkbox, then select a script to execute when the login process fails the test:

- No Remedial Action—Leave Cisco Secure ACS operating as is.

- Reboot—Reboot the system on which Cisco Secure ACS is running.
- Restart All—(Default.) Restart all Cisco Secure ACS services.
- Restart RADIUS/TACACS+—Restart only RADIUS, TACACS+, or both protocols.

You can also develop your own scripts to be executed if there is a system failure. See the online documentation for more information.

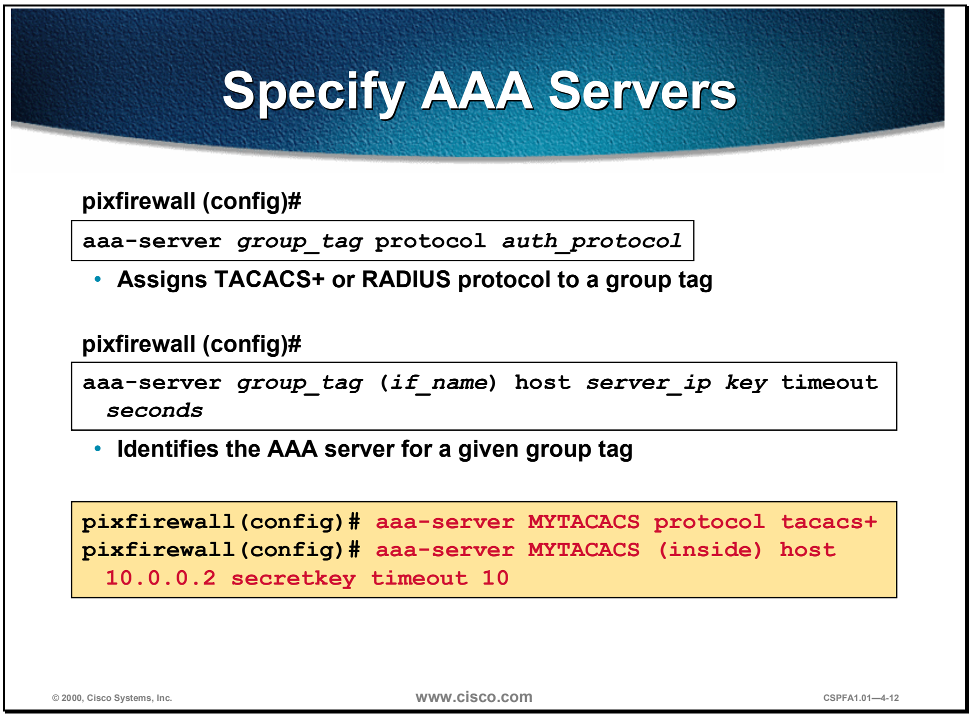
Step 17 To have Cisco Secure ACS generate an e-mail message when administrator events occur, check the **Enable Mail Notifications** checkbox, then enter the following information:

- SMTP Mail Server—The name and domain of the sending mail server; for example, server1.company.com.
- Mail account to notify—The complete e-mail address of the intended recipient; for example, msmith@company.com.

Step 18 Click **Next** The Cisco Sec ure ACS Service Initiation window opens. If you do not want to configure a NAS from Setup, click **Next** To configure a single NAS now, click **Yes, I want to configure Cisco IOS** now. Click **Next**

Authentication Configuration

This section discusses how to configure authentication on the PIX Firewall.



Use the **aaa-server** command to specify AAA server groups. The PIX Firewall lets you define separate groups of TACACS+ or RADIUS servers for specifying different types of traffic, such as a TACACS+ server for inbound traffic and another for outbound traffic. The **aaa** command references the group tag to direct authentication, authorization, or accounting traffic to the appropriate AAA server.

You can have up to 16 tag groups and each group can have up to 16 AAA servers for a total of up to 256 TACACS+ or RADIUS servers. When a user logs in, the servers are accessed one at a time, starting with the first server you specify in the tag group, until a server responds.

The default configuration provides these two aaa-server protocols:

- aaa-server MYTACACS protocol tacacs+
- aaa-server RADIUS protocol radius

Note If you are upgrading from a previous version of PIX Firewall and have aaa command statements in your configuration, using the default server groups lets you maintain backward compatibility with the aaa command statements in your configuration.

Note The previous server type option at the end of the aaa authentication and aaa accounting commands has been replaced with the aaa-server group tag. Backward compatibility with previous versions is maintained by the inclusion of two default protocols for TACACS+ and RADIUS.

Note The PIX Firewall listens for RADIUS on ports 1645 and 1646. If your RADIUS server uses ports 1812 and 1813, you will need to reconfigure it to listen on ports 1645 and 1646.

The syntax for all forms of the **aaa-server** command is as follows:

aaa-server ***group_tag*** **(*if_name*)** **host** ***server_ip key*** **timeout** ***seconds***

no aaa-server ***group_tag*** **(*if_name*)** **host** ***server_ip key*** **timeout** ***seconds***

aaa-server ***group_tag*** **protocol** ***auth_protocol***

clear aaa-server [*group_tag*]

Argument	Description
group_tag	An alphanumeric string that is the name of the server group. Use the ***group_tag*** in the **aaa** command to associate **aaa authentication, aaa authorization, and aaa accounting** command statements to an AAA server.
if_name	The interface name on the side that the AAA server resides.
host *server_ip*	The IP address of the TACACS+ or RADIUS server.
key	A case-sensitive, alphanumeric keyword of up to 127 characters that is the same value as the key on the TACACS+ server. Any characters entered past 127 are ignored. The key is used between the client and server for encrypting data between them. The key must be the same on both the client and server systems. Spaces are not permitted in the key, but other special characters are. If a key is not specified, encryption does not occur.
timeout *seconds*	A retransmit timer that specifies the duration that the PIX Firewall retries access four times to the AAA server before choosing the next AAA server. The default is 5 seconds. The maximum time is 30 seconds. For example, if the timeout value is 10 seconds, PIX Firewall retransmits for 10 seconds and if no acknowledgment is received, tries three times more for a total of 40 seconds to retransmit data before the next AAA server is selected.
protocol *auth_protocol*	The type of AAA server, either TACACS+ or RADIUS.

Enable Authentication

pixfirewall (config)#

```
aaa authentication include|exclude authen_service
  inbound|outbound|if_name local_ip local_mask foreign_ip
  foreign_mask group_tag
```

- **Defines traffic to be authenticated**
- ***authen_service* = any, ftp, http, or telnet**
  - **any: all TCP traffic**

```
pixfirewall(config)# aaa authentication include any inbound
  0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
pixfirewall(config)# aaa authentication include telnet
  outbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
pixfirewall(config)# aaa authentication include ftp dmz
  0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
pixfirewall(config)# aaa authentication exclude any outbound
  10.0.0.33 255.255.255.255 0.0.0.0 0.0.0.0 MYTACACS
```


The **aaa authentication** command enables or disables user authentication services. When you start a connection via Telnet, FTP, or HTTP, you are prompted for a username and password. A AAA server, designated previously with the **aaa-server** command, verifies whether the username and password are correct. If they are correct, the PIX Firewall lets further traffic between the authentication server and the connection interact independently through the PIX Firewall Cut-Through Proxy feature.

The **aaa authentication** command is not intended to mandate your security policy. The AAA servers determine whether a user can or cannot access the system, what services can be accessed, and what IP addresses the user can access. The PIX Firewall interacts with Telnet, FTP, and HTTP to display the prompts for logging. You can specify that only a single service be authenticated, but this must agree with the AAA server to ensure that both the firewall and server agree.

For each IP address, one **aaa authentication** command is permitted for inbound connections and one for outbound connections. The PIX Firewall permits only one authentication type per network. For example, if one network connects through the PIX Firewall using TACACS+ for authentication, another network connecting through the PIX Firewall can authenticate with RADIUS, but one network cannot authenticate with both TACACS+ and RADIUS.

Note The new **include** and **exclude** options are not backward compatible with PIX Firewall versions 5.0 and earlier. If you downgrade to an earlier version, the **aaa authentication** command statements are removed from your configuration.

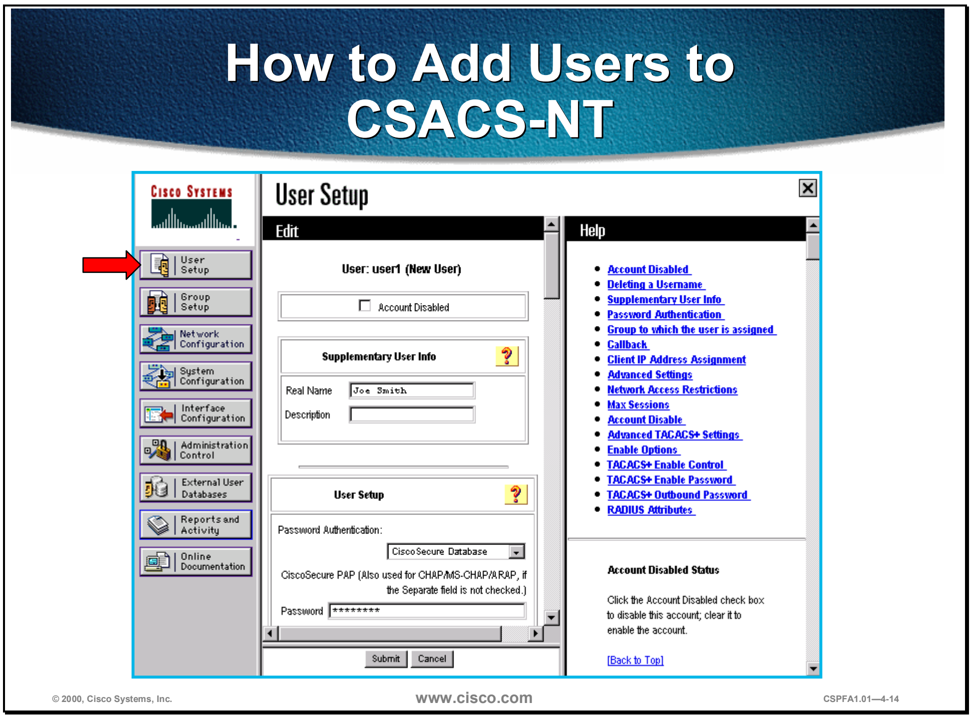
The syntax for all forms of the **aaa authentication** command is as follows:

aaa authentication include | exclude ***authen_service*** **inbound | outbound|*if_name local_ip local_mask foreign_ip foreign_mask group_tag***

no aaa authentication [include | exclude *authen_service* inbound | outbound | *if_name local_ip local_mask foreign_ip foreign_mask group_tag*]

clear aaa [authentication include | exclude *authen_service* inbound | outbound | *if_name local_ip local_mask foreign_ip foreign_mask group_tag*]

Argument	Description
include	Create a new rule with the specified service to include.
exclude	Create an exception to a previously stated rule by excluding the specified service from authentication to the specified host. The **exclude** parameter improves the former except option by allowing the user to specify a port to exclude to a specific host or hosts.
authen_service	The services that require user authentication before they are let through the firewall. Use **any**, **ftp**, **http**, or **telnet**. The **any** value enables authentication for all TCP services.
inbound	Authenticate inbound connections. Inbound means the connection originates on the outside interface and is being directed to the inside or any other perimeter interface.
outbound	Authenticate outbound connections. Outbound means the connection originates on the inside and is being directed to the outside or any other perimeter interface.
if_name	Interface name from which users require authentication. Use ***if_name*** in combination with the ***local_ip*** address and the ***foreign_ip*** address to determine where access is sought and from whom. The ***local_ip*** address is always on the interface with the highest security level and ***foreign_ip*** is always on the lowest.
local_ip	The IP address of the host or network of hosts that you want to be authenticated. You can set this address to 0 to mean all hosts and to let the authentication server decide which hosts are authenticated.
local_mask	Network mask of ***local_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
foreign_ip	The IP address of the hosts you want to access the ***local_ip*** address. Use 0 to mean all hosts.
foreign_mask	Network mask of ***foreign_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
group_tag	The group tag set with the **aaa-server** command.



To add users to the Cisco Secure ACS, complete the following steps:

Step 1 In the navigation bar, click **User Setup**. The Select window opens.

Step 2 Enter a name in the User field.

Note The username can contain up to 32 characters. Names cannot contain the following special characters: #?"*><. Leading and trailing spaces are not allowed.

Step 3 Click **Add/Edit**. The Edit window opens. The username being added or edited appears at the top of the window.

Account Disable

Click the Account Disabled check box to deny access for this user.

Note You must click **Submit** to have this action take effect.

Supplementary User Information

- Supplementary User Information—(Optional.) Enter the following information:
  - Real Name—If the username is not the user's real name, enter the real name here.
  - Description—Enter a detailed description of the user.

Note This item can contain up to five user-configurable fields. See the "Interface Configuration" section for information on how to display and configure these fields.

User Setup

Edit or enter the following information for the user as applicable:

- Password Authentication—Select the authentication type from the drop-down menu:
  - Cisco Secure Database—Authenticates a user from the local Cisco Secure ACS database.
  - Windows NT—Authenticates a user with an existing account in the Windows NT User Database located on the same machine as the Cisco Secure server. There is also an entry in the Cisco Secure ACS database used for other Cisco Secure ACS services. This authentication type will appear in the user interface only if this external user database has been configured in External User Databases: Database Configuration.
- Password and Confirm Password—Enter and confirm the Password Authentication Protocol (PAP) password to be used.
- Separate CHAP/MS-CHAP/ARAP—This is not used with the PIX Firewall.

Note The Password and Confirm Password fields are required for all authentication methods except for all third-party user databases.

- Group to which the user is assigned—From the drop-down menu, select the group to which to assign the user. The user inherits the attributes and operations assigned to the group. By default, users are assigned to the Default Group. Users who authenticate via the Unknown User method who are not found in an existing group are also assigned to the Default Group.
- Callback—This is not used with the PIX Firewall.
- Client IP Address Assignment—This is not used with PIX Firewall.

Account Disable

Define the circumstances under which this user's account will become disabled.

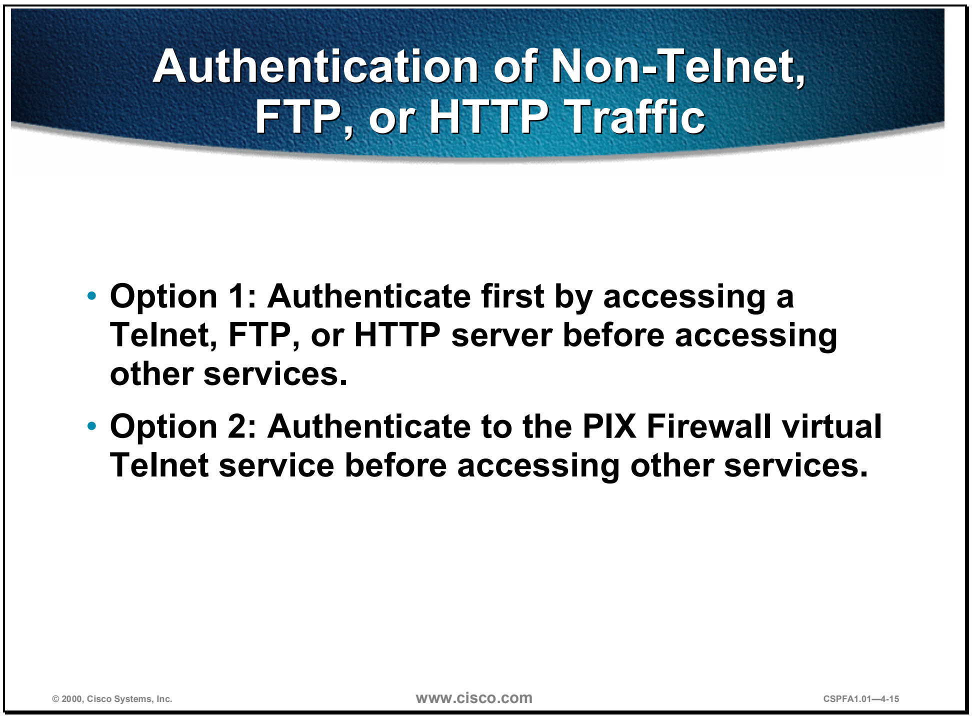
Note This is not to be confused with account expiration due to Password Aging. Password Aging is defined for groups only, not for individual users.

- Never—Click to keep the user's account always enabled. This is the default.
- Disable account if—Click to disable the account under the circumstances you specify in the following fields:
  - Date exceeds—From the drop-down menus, select the month, date, and year on which to disable the account. The default is 30 days after the user is added.

- Failed attempts exceed—Click the check box and enter the number of consecutive unsuccessful login attempts to allow before disabling the account. The default is 5.
- Failed attempts since last successful login—This counter shows the number of unsuccessful login attempts since the last time this user logged in successfully.
  - Reset current failed attempts count on submit—If an account is disabled because the failed attempts count has been exceeded, check this check box and click **Submit** to reset the failed attempts counter to 0 and reinstate the account.

If you are using the Windows NT user database, this expiration information is in addition to the information in the Windows NT user account. Changes here do not alter settings configured in Windows NT.

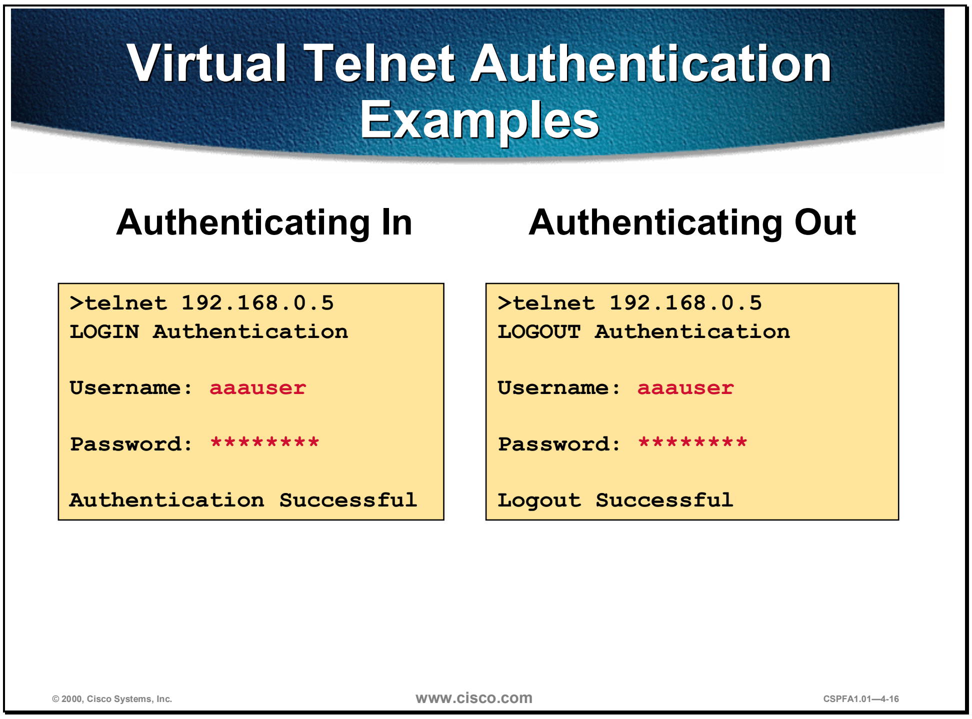
When you have finished configuring all user information, click **Submit**.



The PIX Firewall authenticates users via Telnet, FTP, or HTTP. But what if users need to access a Microsoft file server on port 139 or a Cisco IP/TV server for instance? Whenever users are required to authenticate to access services other than Telnet, FTP, or HTTP, they need to do one of the following:

- Option 1: Authenticate first by accessing a Telnet, FTP, or HTTP server before accessing other services.
- Option 2: Authenticate to the PIX Firewall virtual Telnet service before accessing other services.

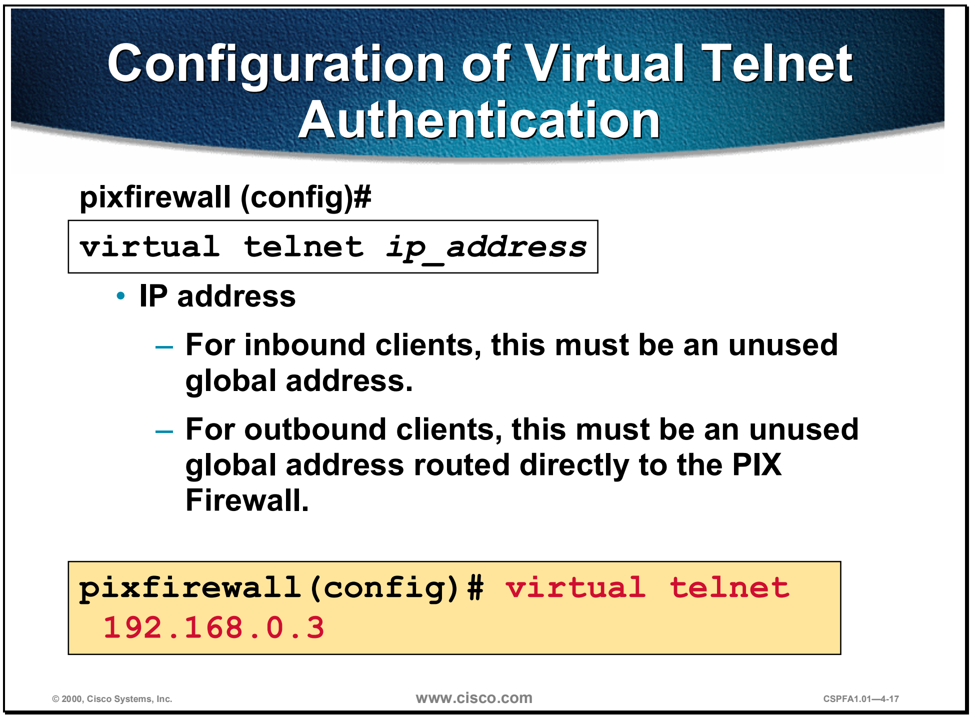
When there are no Telnet, FTP, or HTTP servers to authenticate with, or just to simplify authentication for the user, the PIX Firewall allows a virtual Telnet authentication option. This permits the user to authenticate directly with the PIX Firewall to the virtual Telnet IP address.



The virtual Telnet option provides a way to pre-authenticate users who require connections through the PIX Firewall using services or protocols that do not support authentication. The virtual Telnet IP address is used both to authenticate in and authenticate out of the PIX Firewall.

When an unauthenticated user Telnets to the virtual IP address, the user is challenged for their username and password, and then authenticated with the TACACS+ or RADIUS server. Once authenticated, the user sees the message "Authentication Successful" and the authentication credentials are cached in the PIX Firewall for the duration of the uauth timeout.

If a user wishes to log out and clear the entry in the PIX Firewall uauth cache, the user can again Telnet to the virtual address. The user is prompted for a username and password, the PIX Firewall removes the associated credentials from the uauth cache, and the user receives a "Logout Successful" message.



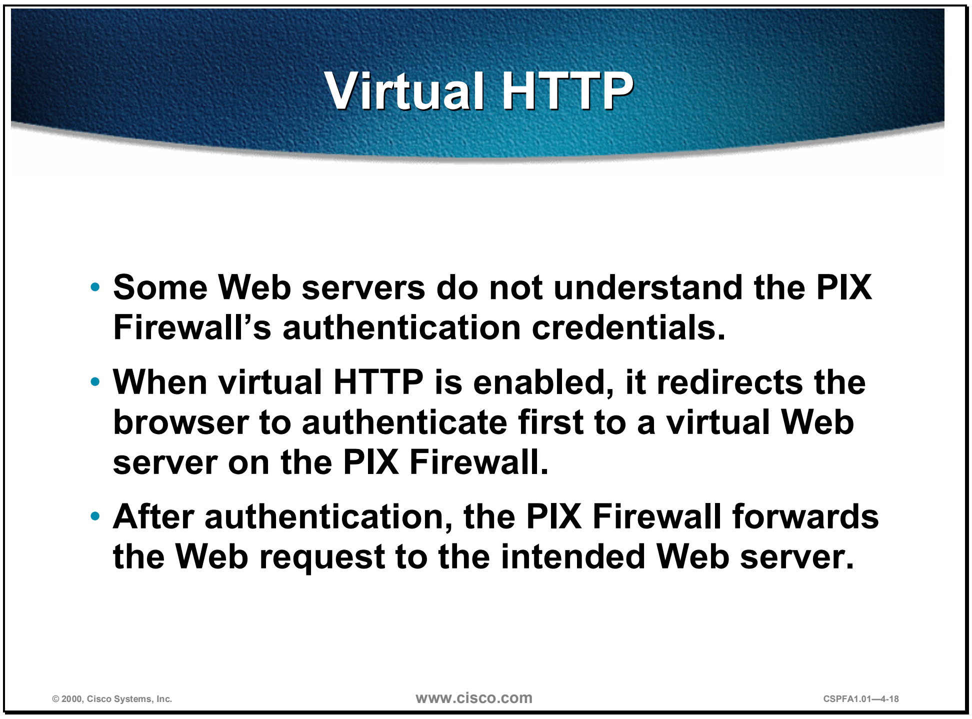
When using virtual Telnet to authenticate inbound clients, the IP address must be an unused global address.

When using virtual Telnet to authenticate outbound clients, this must be an unused global address routed directly to the PIX Firewall.

The syntax for the **virtual telnet** command is as follows:

virtual telnet *ip_address*

Argument	Description
ip_address	Unused global IP address on PIX Firewall, used for Telnet for authentication.

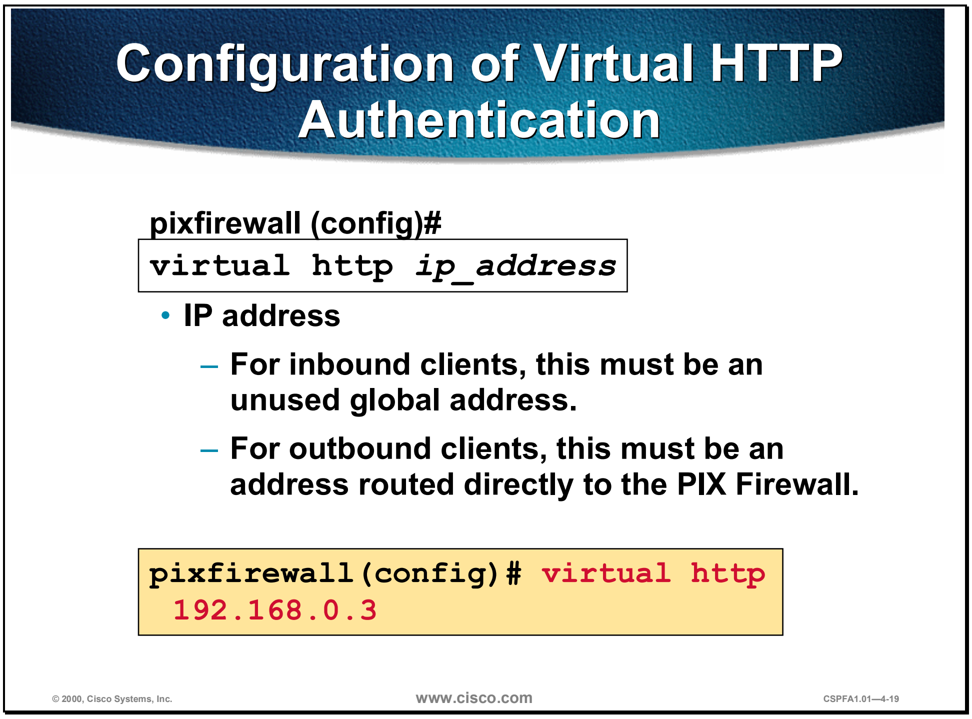


With the virtual HTTP option, web browsers work correctly with the PIX Firewall’s HTTP authentication. The PIX Firewall assumes that the AAA server database is shared with a web server and automatically provides the AAA server and web server with the same information. The virtual HTTP option works with the PIX Firewall to authenticate the user, separate the AAA server information from the web client’s URL request, and direct the web client to the web server. The virtual HTTP option works by redirecting the web browser’s initial connection to an IP address, which resides in the PIX Firewall, authenticating the user, then redirecting the browser back to the URL that the user originally requested. This option is so named because it accesses a virtual HTTP server on the PIX Firewall, which in reality does not exist.

This option is especially useful for PIX Firewall interoperability with Microsoft IIS, but is useful for other authentication servers. When using HTTP authentication to a site running Microsoft IIS that has “Basic text authentication” or “NT Challenge” enabled, users may be denied access from the Microsoft IIS server. This occurs because the browser appends the string: “Authorization: Basic=Uuhjksdkfhk==” to the HTTP GET commands. This string contains the PIX Firewall authentication credentials. Windows NT Microsoft IIS servers respond to the credentials and assume that a Windows NT user is trying to access privileged pages on the server. Unless the PIX Firewall username and password combination is exactly the same as a valid Windows NT username and password combination on the Microsoft IIS server, the HTTP GET command is denied.

To solve this problem, PIX Firewall redirects the browser’s initial connection to its virtual HTTP IP address, authenticates the user, then redirects the browser back to the URL that the user originally requested.

Note Do not set the timeout uauth duration to 0 seconds when using the virtual HTTP option. This will prevent HTTP connections to the real web server.

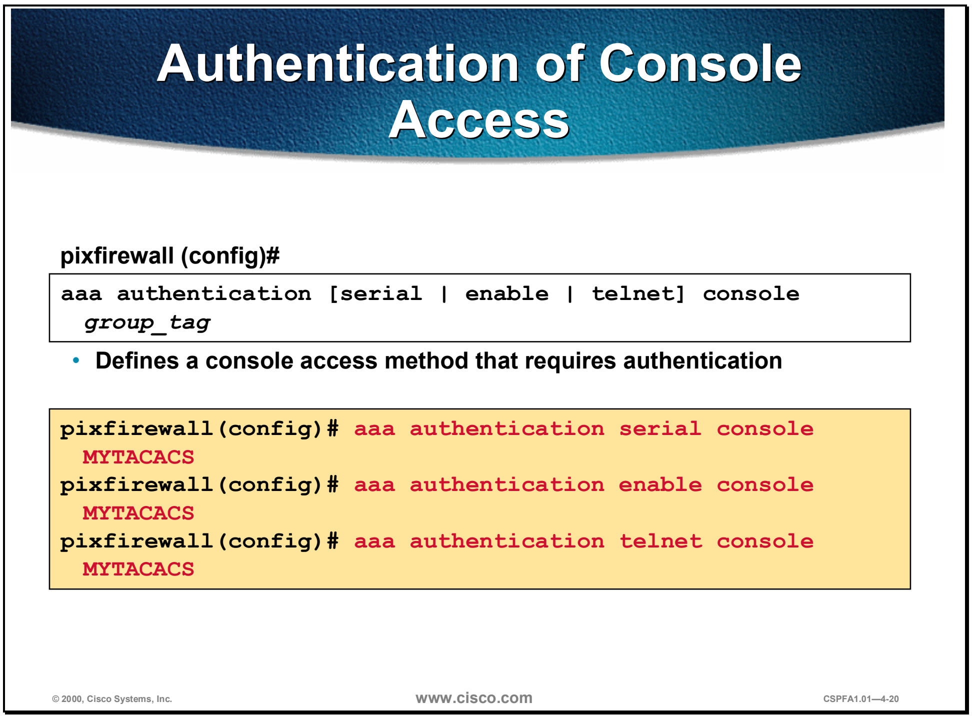


The syntax for the **virtual http** command is as follows:

virtual http ***ip_address*** **[warn]**

no virtual http ***ip_address***

Argument	Description
ip_address	PIX Firewall's network interface IP address.
warn	Informs **virtual http** command users that the command was redirected. This option is only applicable for text-based browsers where the redirect cannot happen automatically.



Use the **aaa authentication console** command to require authentication verification to access the PIX Firewall's serial, enable, or Telnet consoles. The serial console options also log to a Syslog server change made to the configuration from the serial console.

Authenticated access to the PIX Firewall console has different types of prompts depending on the option you choose. While the **enable** option allows three tries before stopping with an access denied message, both the **serial** and **Telnet** options cause you to be prompted continually until you have successfully logged in.

The **serial** option requests a username and password before the first command-line prompt on the serial console connection. The **telnet** option forces you to specify a username and password before the first command-line prompt of a Telnet console connection. The **enable** option requests a username and password before accessing privileged mode for serial or Telnet connections.

Telnet access to the PIX Firewall console is available from any internal interface (not the outside interface) and requires previous use of the **telnet** command.

Authentication of the serial console creates a potential dead-lock situation if the authentication server requests are not answered and you need access to the console to attempt diagnosis. If the console login request times out, you can gain access to the PIX Firewall from the serial console by entering the PIX Firewall username and the enable password.

The maximum password length for accessing the console is 16 characters.

The syntax for the **aaa authentication console** command is as follows:

aaa authentication [serial | enable | telnet] console ***group_tag***

no aaa authentication [serial | enable | telnet] console ***group_tag***

Argument	Description
serial	Requests a username and password before the first command-line prompt on the serial console connection.
enable	Requests a username and password before accessing privileged mode for serial or Telnet connections.
telnet	Forces you to specify a username and password before the first command-line prompt of a Telnet console connection.
console	Specifies that access to the PIX Firewall console requires authentication and, as an option, logs configuration changes to a syslog server.
group_tag	The group tag set with the **aaa-server** command.

How to Change the Authentication Timeouts

pixfirewall (config)#

```
timeout uauth hh:mm:ss [absolute|inactivity]
```

- Sets the time interval before users will be required to reauthenticate
  - Absolute: Time interval starts at user login
  - Inactivity: Time interval for inactive sessions (no traffic)

```
pixfirewall(config)# timeout uauth 3:00:00 absolute
pixfirewall(config)# timeout uauth 0:30:00 inactivity
```

 www.cisco.com CSPFA1.01—4-21

Use the **timeout uauth** command to specify how long the cache should be kept after the user connections become idle. The timeout command value must be at least 2 minutes. Use the **clear uauth** command to delete all authorization caches for all users, which will cause them to have to reauthenticate the next time they create a connection.

The **inactivity** and **absolute** qualifiers cause users to have to reauthenticate after either a period of inactivity or an absolute duration. The inactivity timer starts after a connection becomes idle. If a user establishes a new connection before the duration of the inactivity timer, the user is not required to reauthenticate. If a user establishes a new connection after the inactivity timer expires, the user must reauthenticate.

The absolute timer runs continuously, but waits to reprompt the user when the user starts a new connection, such as clicking a link after the absolute timer has elapsed, then the user is prompted to reauthenticate. The absolute timer must be shorter than the xlate timer; otherwise, a user could be reprompted after their session already ended.

The inactivity timer give users the best Internet access because they are not prompted to regularly reauthenticate. Absolute timers provide security and manage the PIX Firewall connections better. By being prompted to reauthenticate regularly, users manage their use of the resources more efficiently. Also by being reprompted, you minimize the risk that someone will attempt to use another user's access after they leave their workstation, such as in a college computer lab. You may want to set an absolute timer during peak hours and an inactivity timer during other times.

Both an inactivity timer and an absolute timer can operate at the same time, but you should set the absolute timer duration for longer than the inactivity timer. If the absolute timer is less than the inactivity timer, the inactivity timer never occurs. For example, if you set the absolute timer to 10 minutes and the inactivity timer to an hour, the absolute timer reprompts the user every 10 minutes, and the inactivity timer will never be started.

If you set the inactivity timer to some duration, but the absolute timer to zero, then users are only reauthenticated after the inactivity timer elapses. If you set both timers to zero, then users have to reauthenticate on every new connection.

Note Do not set the **timeout uauth** duration to 0 seconds when using the virtual HTTP option or passive FTP.

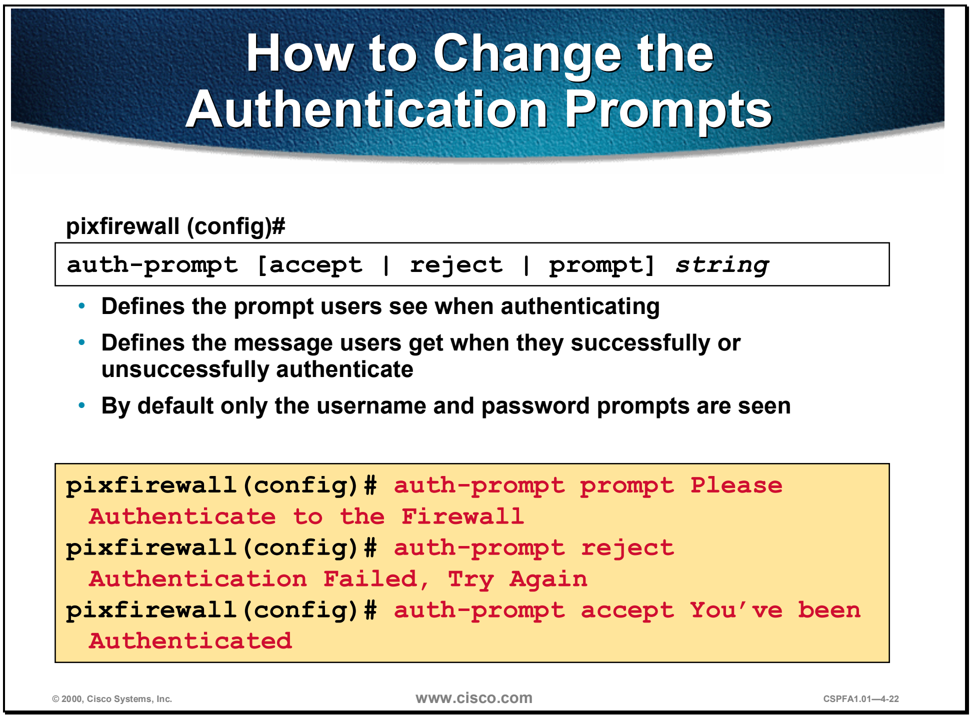
The syntax for the **timeout uauth** command is as follows:

timeout uauth [*hh:mm:ss*] [absolute | inactivity]

show timeout

clear uauth

Argument	Description
uauth *hh:mm:ss*	Duration before the authentication and authorization cache times out and user has to re-authenticate next connection. This duration must be shorter than the xlate values. Set to 0 to disable caching.
absolute	Run uauth timer continuously, but after timer elapses, wait to reprompt the user until the user starts a new connection, such as clicking a link in a web browser. To disable absolute, set it to zero (0). Default is 5 minutes.
inactivity	Start uauth timer after a connection becomes idle. Default is 0.



Use the **auth-prompt** command to change the AAA challenge text for HTTP, FTP, and Telnet access. This text displays above the username and password prompts that you view when logging in.

Note Microsoft Internet Explorer only displays up to 37 characters in an authentication prompt, Netscape Navigator displays up to 120 characters, and Telnet and FTP display up to 235 characters in an authentication prompt.

The syntax for the **auth-prompt** command is as follows:

auth-prompt [accept | reject | prompt] ***string***

no auth-prompt [accept | reject | prompt] ***string***

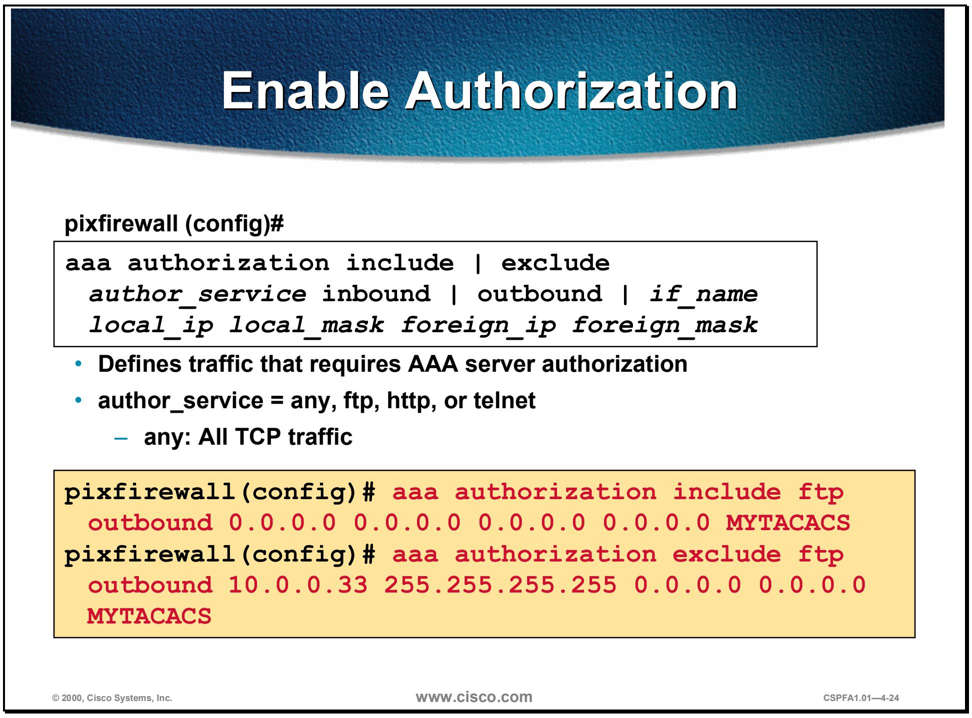
show auth-prompt

clear auth-prompt

Argument	Description
accept	If a user authentication via Telnet is accepted, the accept message is displayed.
reject	If a user authentication via Telnet is rejected, the reject message is displayed.
prompt	The AAA challenge prompt string follows this keyword. This keyword is optional for backward compatibility.
string	A string of up to 235 alphanumeric characters. Special characters should not be used; however, spaces and punctuation characters are permitted. Entering a question mark or pressing the Enter key ends the string. (The question mark appears in the string.)

Authorization Configuration

This section discusses the configuration of the PIX Firewall for authorization.



The PIX Firewall uses authorization services with TACACS+ AAA servers that determine which services an authenticated user can access.

Note The PIX Firewall does not support RADIUS authorization.

The syntax for the **aaa authorization** command is as follows:

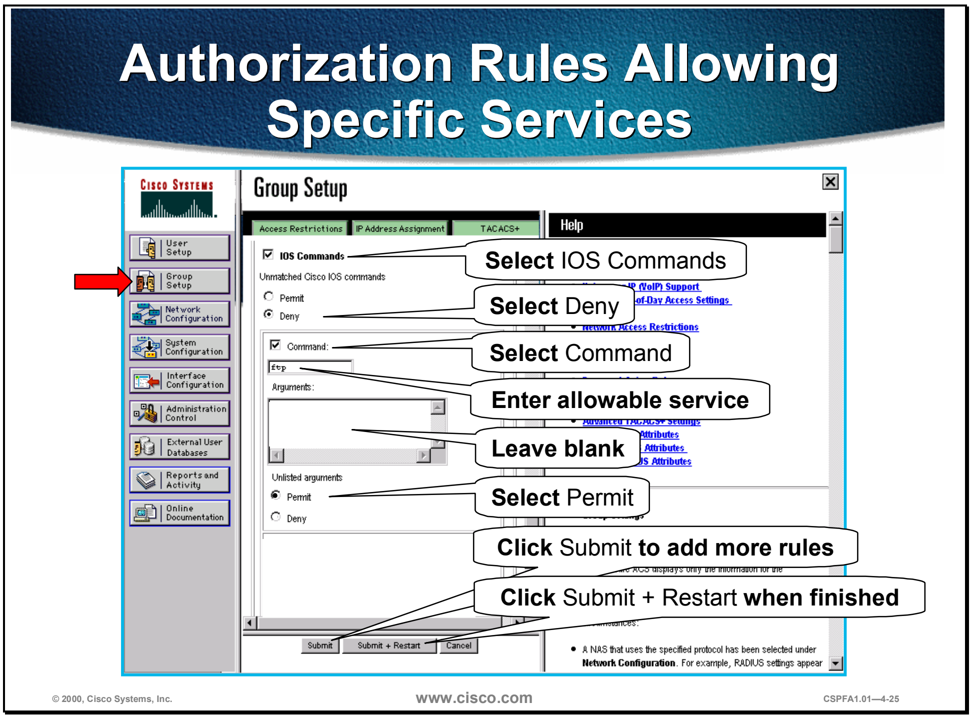
aaa authorization include | **exclude** ***author_service*** **inbound** | **outbound** | ***if_name local_ip local_mask foreign_ip foreign_mask***

no aaa authorization [**include** | **exclude** ***author_service*** **inbound** | **outbound** | ***if_name local_ip local_mask foreign_ip foreign_mask***]

clear aaa [**authorization** [**include** | **exclude** ***author_service*** **inbound** | **outbound** | ***if_name local_ip local_mask foreign_ip foreign_mask***]]

Argument	Description
include *author_service*	The services that require authorization. Use **any**, **ftp**, **http**, or **telnet**. Services not specified are authorized implicitly. Services specified in the **aaa authentication** command do not affect the services which require authorization.

exclude *author_service*	Create an exception to a previously stated rule by excluding the specified service from authorization to the specified host or networks.
inbound	Authenticate or authorize inbound connections. Inbound means the connection originates on the outside interface and is being directed to the inside or any other perimeter interface.
outbound	Authenticate or authorize outbound connections. Outbound means the connection originates on the inside and is being directed to the outside or any other perimeter interface.
if_name	Interface name from which users require authentication. Use ***if_name*** in combination with the ***local_ip*** address and the ***foreign_ip*** address to determine where access is sought and from whom.
local_ip	The IP address of the host or network of hosts that you want to be authenticated or authorized. You can set this address to 0 to mean all hosts and to let the authentication server decide which hosts are authenticated.
local_mask	Network mask of ***local_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
foreign_ip	The IP address of the hosts you want to access the ***local_ip*** address. Use 0 to mean all hosts.
foreign_mask	Network mask of ***foreign_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.



Complete the following steps to add authorization rules for specific services in Cisco Secure ACS:

Step 1 In the navigation bar, click **Group Setup**. The Group Setup window opens.

Step 2 Scroll down in Group Setup until you find IOS Commands.

Step 3 Select **IOS Commands**.

Step 4 Under Unmatched Cisco IOS commands, select **Deny**

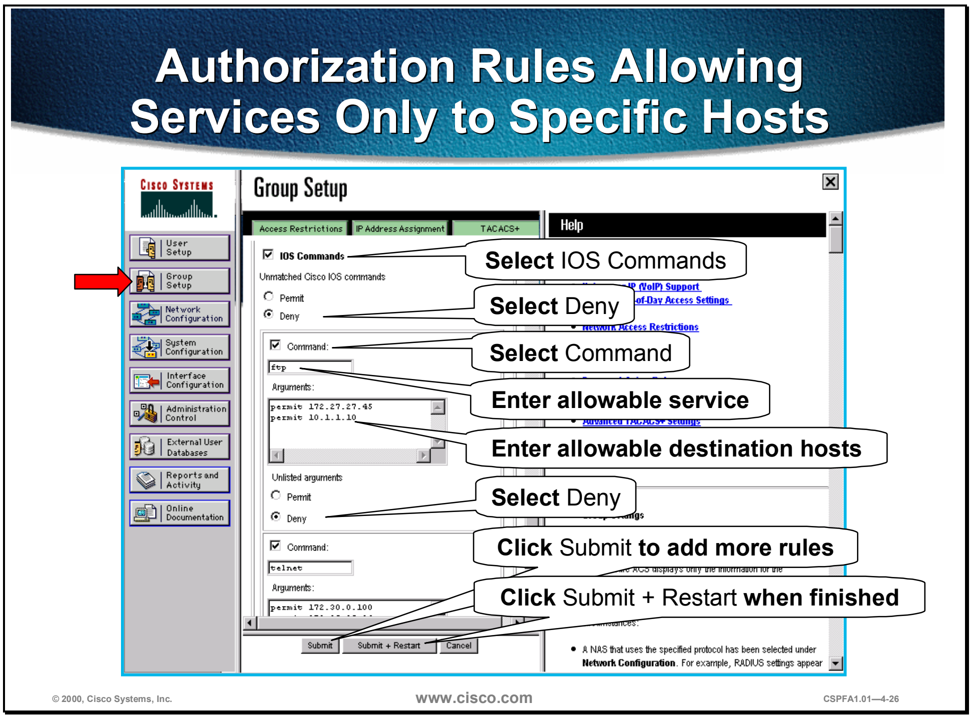
Step 5 Select **Command**.

Step 6 Enter the allowable service: **ftp**, **telnet**, or **http**.

Step 7 Leave the **Arguments** field blank.

Step 8 Under Unlisted arguments, select **Permit**.

Step 9 Click **Submit** to add more rules, or click **Submit + Restart** when finished.



Complete the following steps to add authorization rules for services to specific hosts in Cisco Secure ACS:

Step 1 In the navigation bar, click **Group Setup**. The Group Setup window opens.

Step 2 Scroll down in Group Setup until you find IOS Commands.

Step 3 Select **IOS Commands**.

Step 4 Under Unmatched Cisco IOS commands, select **Deny**

Step 5 Select **Command**.

Step 6 Enter the allowable service: **ftp**, **telnet**, or **http**.

Step 7 In the Arguments field, enter the IP addresses of the host that users are authorized to go to. Use the following format:

permit *ip_addr* (where ***ip_addr*** is the IP address 6the host)

Step 8 Under Unlisted arguments, select **Deny**

Step 9 Click **Submit** to add more rules, or click **Submit + Restart** when finished.

Authorization of Non-Telnet, FTP, or HTTP Traffic

pixfirewall (config)#

```
aaa authorization include | exclude author_service inbound |
  outbound | if_name local_ip local_mask foreign_ip foreign_mask
```

- author_service = protocol/port
  - protocol: tcp (6), udp (17), icmp (1), or others (protocol #)
  - port:
    - single port (e.g., 53), port range (e.g., 2000-2050), or port 0 (all ports)
    - ICMP message type (8 = echo request, 0 = echo reply)
    - port is not used for protocols other than TCP, UDP, or ICMP

```
pixfirewall(config)# aaa authorization include udp/0 inbound
  0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
pixfirewall(config)# aaa authorization include tcp/30-100 outbound
  0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
pixfirewall(config)# aaa authorization include icmp/8 outbound
  0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
```

 www.cisco.com CSPFA1.01—4-27

The syntax of the **aaa authorization** of non-Telnet, FTP, or HTTP command is as follows:

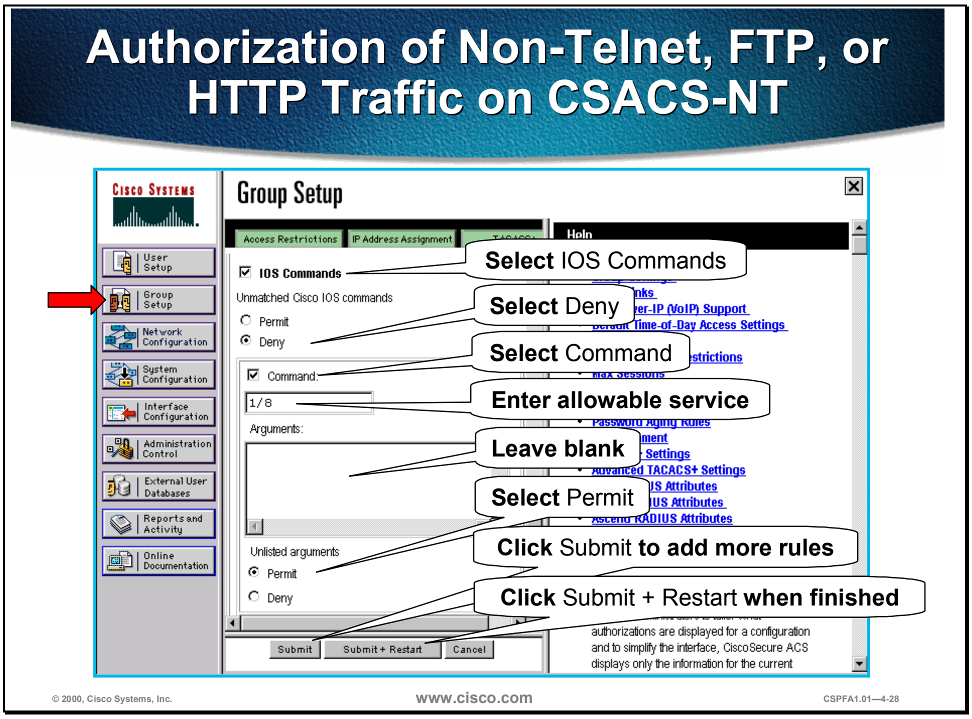
aaa authorization include | **exclude** *author_service* **inbound** | **outbound** | *if_name local_ip local_mask foreign_ip foreign_mask*

no aaa authorization [**include** | **exclude** *author_service* **inbound** | **outbound** | *if_name local_ip local_mask foreign_ip foreign_mask*]

clear aaa [**authorization** [**include** | **exclude** *author_service* **inbound** | **outbound** | *if_name local_ip local_mask foreign_ip foreign_mask*]]

Argument	Description
include ***author_service***	The services which require authorization. Use *protocol* or *port*. Services not specified are authorized implicitly. Services specified in the **aaa authentication** command do not affect the services that require authorization.
exclude ***author_service***	Create an exception to a previously stated rule by excluding the specified service from authorization to the specified host or networks.
inbound	Authenticate or authorize inbound connections. Inbound means the connection originates on the outside interface and is being directed to the inside or any other perimeter interface.
outbound	Authenticate or authorize outbound connections. Outbound means the connection originates on the inside and is being directed to the outside or any other perimeter interface.

Argument	Description
if_name	Interface name from which users require authentication. Use ***if_name*** in combination with the ***local_ip*** address and the ***foreign_ip*** address to determine where access is sought and from whom.
local_ip	The IP address of the host or network of hosts that you want to be authenticated or authorized. You can set this address to 0 to mean all hosts and to let the authentication server decide which hosts are authenticated.
local_mask	Network mask of ***local_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
foreign_ip	The IP address of the hosts you want to access the ***local_ip*** address. Use 0 to mean all hosts.
foreign_mask	Network mask of ***foreign_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.



Complete the following steps to add authorization rules for specific non-telnet, FTP, or HTTP services in Cisco Secure ACS:

Step 1 In the navigation bar, click **Group Setup**. The Group Setup window opens.

Step 2 Scroll down in Group Setup until you find IOS Commands.

Step 3 Select **IOS Commands**.

Step 4 Under Unmatched Cisco IOS commands, select **Deny**

Step 5 Select **Command**.

Step 6 Enter an allowable service using the following format: **protocol** or **port** (where *protocol* is the protocol number and *port* is the port number).

Step 7 Leave the **Arguments** field blank.

Step 8 Under Unlisted arguments, select **Permit**.

Step 9 Click **Submit** to add more rules, or click **Submit + Restart** when finished.

Accounting Configuration

This section demonstrates how to enable and configure accounting for all services, select services, or no services.

Enable Accounting

pixfirewall (config)#

```
aaa accounting include | exclude acctg_service
  inbound | outbound | if_name local_ip
  local_mask foreign_ip foreign_mask group_tag
```

- Defines traffic that requires AAA server accounting
- *acctg_service* = any, ftp, http, or telnet
  - any: All TCP traffic

```
pixfirewall(config)# aaa accounting include any
  outbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
pixfirewall(config)# aaa accounting exclude any
  outbound 10.0.0.33 255.255.255.255 0.0.0.0 0.0.0.0
  MYTACACS
```

 www.cisco.com CSPFA1.01—4-30

The syntax for the **aaa accounting** command is as follows:

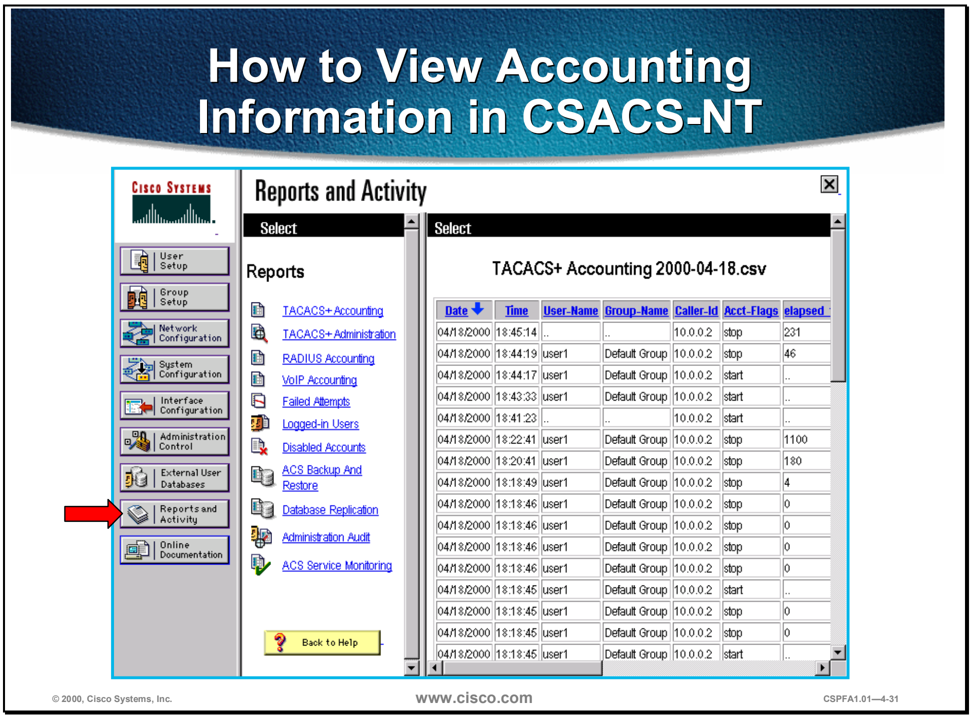
aaa accounting include | **exclude** *acctg_service* **inbound** | **outbound** | *if_name local_ip local_mask foreign_ip foreign_mask group_tag*

no aaa accounting include | **exclude** *authen_service* **inbound** | **outbound** | *if_name group_tag*

clear aaa [**accounting include** | **exclude** *authen_service* **inbound** | **outbound** | *if_name group_tag*]

Argument	Description
include ***acctg_service***	The accounting service. Accounting is provided for all services, or you can limit it to one or more services. Possible values are **any**, **ftp**, **http**, or **telnet**. Use **any** to provide accounting for all TCP services. To provide accounting for UDP services, use the *protocol/port* form.
exclude ***acctg_service***	Create an exception to a previously stated rule by excluding the specified service from authentication, authorization, or accounting to the specified host. The **exclude** parameter improves the former **except** option by allowing the user to specify a port to exclude to a specific host or hosts.

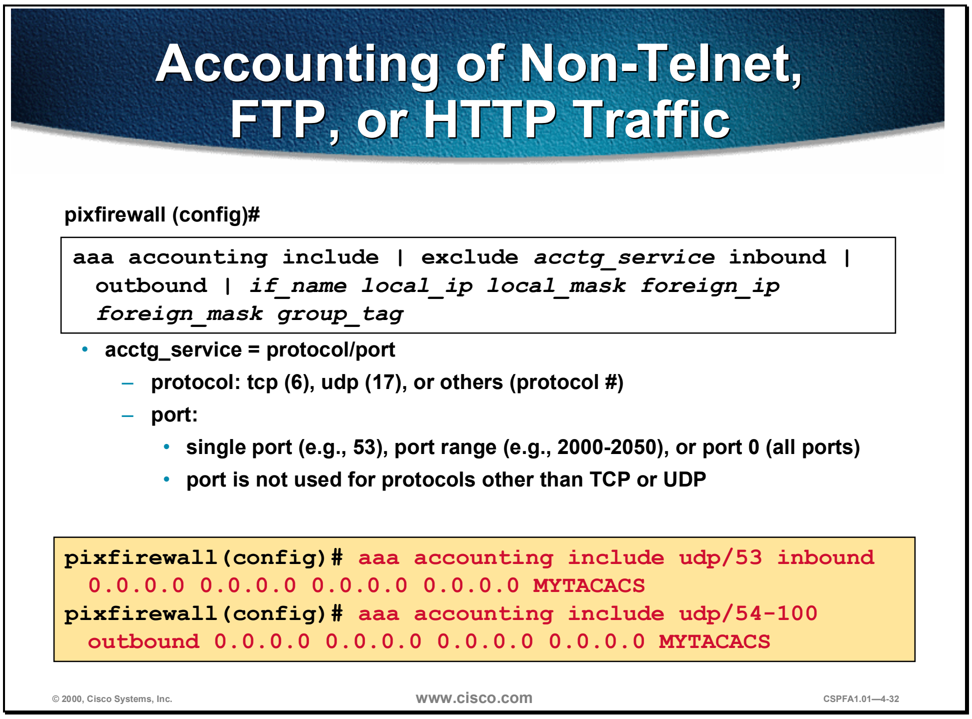
Argument	Description
inbound	Authenticate or authorize inbound connections. Inbound means the connection originates on the outside interface and is being directed to the inside or any other perimeter interface.
outbound	Authenticate or authorize outbound connections. Outbound means the connection originates on the inside and is being directed to the outside or any other perimeter interface.
if_name	Interface name from which users require authentication. Use ***if_name*** in combination with the ***local_ip*** address and the ***foreign_ip_address*** to determine where access is sought and from whom.
local_ip	The IP address of the host or network of hosts that you want to be authenticated or authorized. You can set this address to 0 to mean all hosts and to let the authentication server decide which hosts are authenticated.
local_mask	Network mask of ***local_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
foreign_ip	The IP address of the hosts you want to access the ***local_ip*** address. Use 0 to mean all hosts.
foreign_mask	Network mask of ***foreign_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
group_tag	The group tag set with the **aaa-server** command.



Complete the following steps to add authorization rules for specific non-telnet, FTP, or HTTP services in Cisco Secure ACS:

Step 1 In the navigation bar, click **Reports and Activity**. The Report and Activity window opens.

Step 2 Under Reports, click **TACACS+ Accounting** to display the accounting records.



The syntax for the **aaa accounting** of non-Telnet, FTP, or HTTP traffic command is as follows:

aaa accounting include | exclude ***acctg_service*** **inbound | outbound |** ***if_name local_ip local_mask foreign_ip foreign_mask group_tag***

no aaa accounting include | exclude authen_service inbound | outbound | if_name group_tag

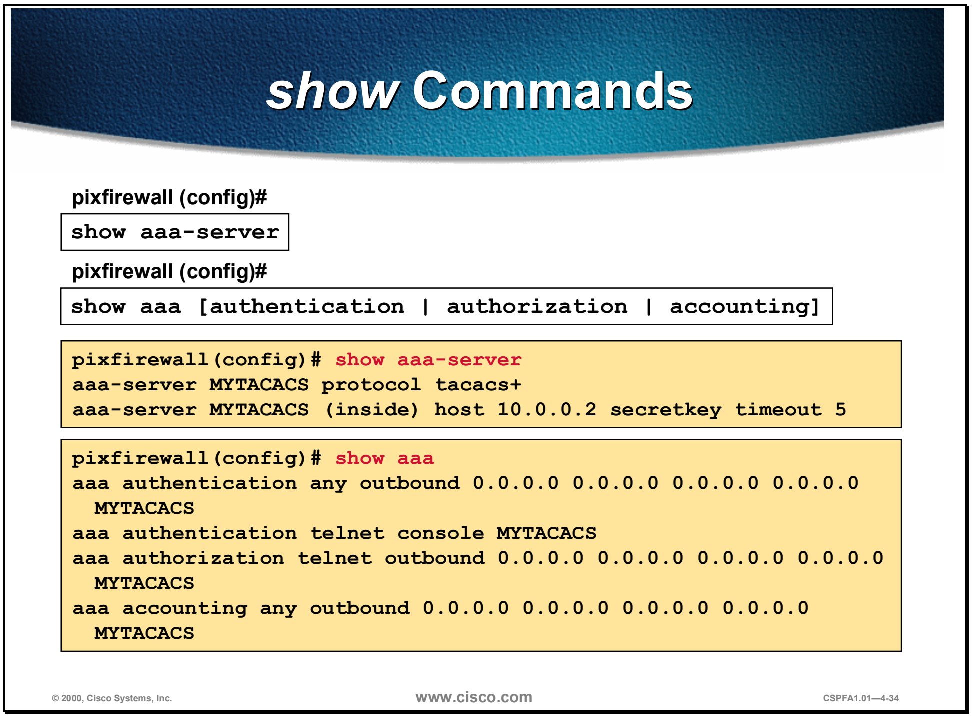
clear aaa [accounting include | exclude ***authen_service*** **inbound | outbound |** ***if_name group_tag*****]**

Argument	Description
include ***acctg_service***	The accounting service. Accounting is provided for all services or you can limit it to one or more services. Possible values are **any**, **ftp**, **http**, or **telnet**. Use **any** to provide accounting for all TCP services. To provide accounting for UDP services, use the *protocol/port* form.
exclude ***acctg_service***	Create an exception to a previously stated rule by excluding the specified service from authentication, authorization, or accounting to the specified host. The **exclude** parameter improves the former except option by allowing the user to specify a port to exclude to a specific host or hosts.
inbound	Authenticate or authorize inbound connections. Inbound means the connection originates on the outside interface and is being directed to the inside or any other perimeter interface.
outbound	Authenticate or authorize outbound connections. Outbound means the connection originates on the inside and is being directed to the outside or any other perimeter interface.

Argument	Description
if_name	Interface name from which users require authentication. Use ***if_name*** in combination with the ***local_ip*** address and the ***foreign_ip*** address to determine where access is sought and from whom.
local_ip	The IP address of the host or network of hosts that you want to be authenticated or authorized. You can set this address to 0 to mean all hosts and to let the authentication server decide which hosts are authenticated.
local_mask	Network mask of ***local_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
foreign_ip	The IP address of the hosts you want to access the ***local__ip*** address. Use 0 to mean all hosts.
foreign_mask	Network mask of ***foreign_ip***. Always specify a specific mask value. Use 0 if the IP address is 0. Use 255.255.255.255 for a host.
group_tag	The group tag set with the **aaa-server** command.

Troubleshooting the AAA Configuration

This section discusses the procedure for verifying the AAA configuration.



The syntax for the **show aaa-server** and **show aaa** commands are as follows:

show aaa-server

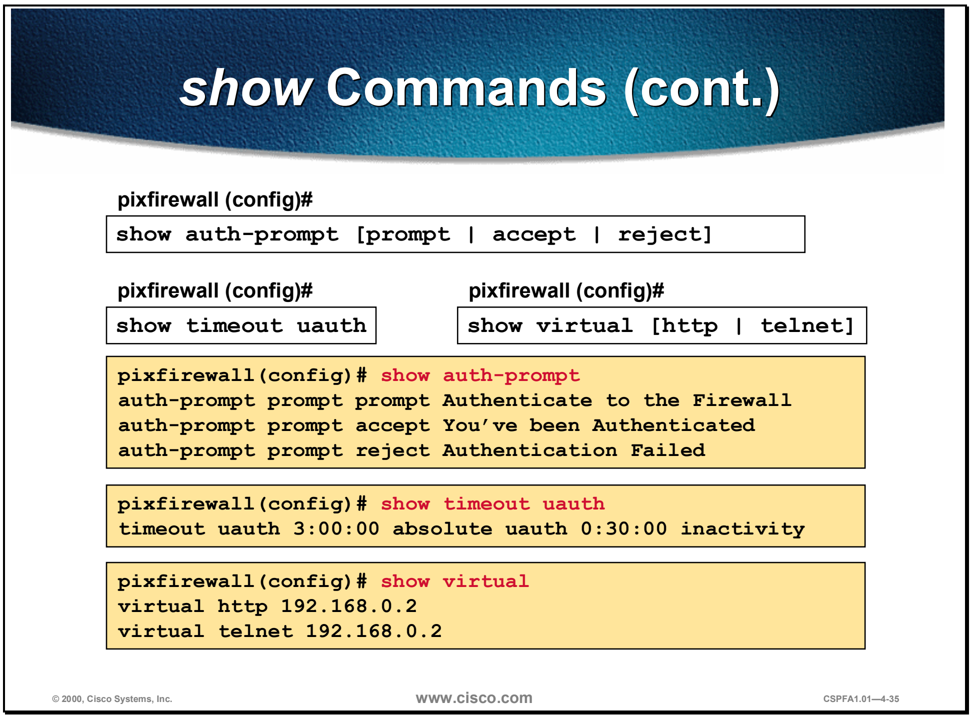
clear aaa-server [group_tag]

no aaa-server ***group_tag*** **(*if_name*) host** ***server_ip key*** **timeout** ***seconds***

show aaa [authentication | authorization | accounting]

Argument	Description
group tag	An alphanumeric string that is the name of the server group.
if_name	The interface name on which the server resides.
host ***server_ip***	The IP address of the TACACS+ or RADIUS server.
key	A case-sensitive, alphanumeric keyword of up to 127 characters that is the same value as the key on the TACACS+ server. Any characters entered past 127 are ignored. The key is used between the client and server for encrypting data between them. The key must be the same on both the client and server systems. Spaces are not permitted in the key, but other special characters are.

Argument	Description
timeout *seconds*	A retransmit timer that specifies the duration that the PIX Firewall retries access. Access is retried four times to the AAA server before choosing the next AAA server. The default is 5 seconds. The maximum time is 30 seconds.
authentication	Displays user authentication, prompts user for username and password, and verifies information with the authentication server.
authorization	Displays TACACS+ user authorization for services. (PIX Firewall does not support RADIUS authorization.) The authentication server determines what services the user is authorized to access.
accounting	Displays accounting services with authentication server. Use of this command requires that you previously used the **aaa-server** command to designate an authentication server.



The syntax for the **show auth-prompt**, **show timeout uauth**, and the **show virtual** commands are as follows:

show auth-prompt [**prompt** | **accept** | **reject**]

show timeout uauth

show virtual [**http** | **telnet**]

Argument	Description
prompt	Displays the prompt users get when authenticating.
accept	Displays the message users get when successfully authenticating.
reject	Displays the message users get when unsuccessfully authenticating.
timeout uauth	Displays the current uauth timer values for all authenticated users.
http	Displays the virtual HTTP configuration.
telnet	Displays the virtual Telnet configuration.

Lab: Configure AAA on the Cisco Secure PIX Firewall Using Cisco Secure ACS for Windows NT

Complete the following lab exercises to practice what you have learned in this chapter.

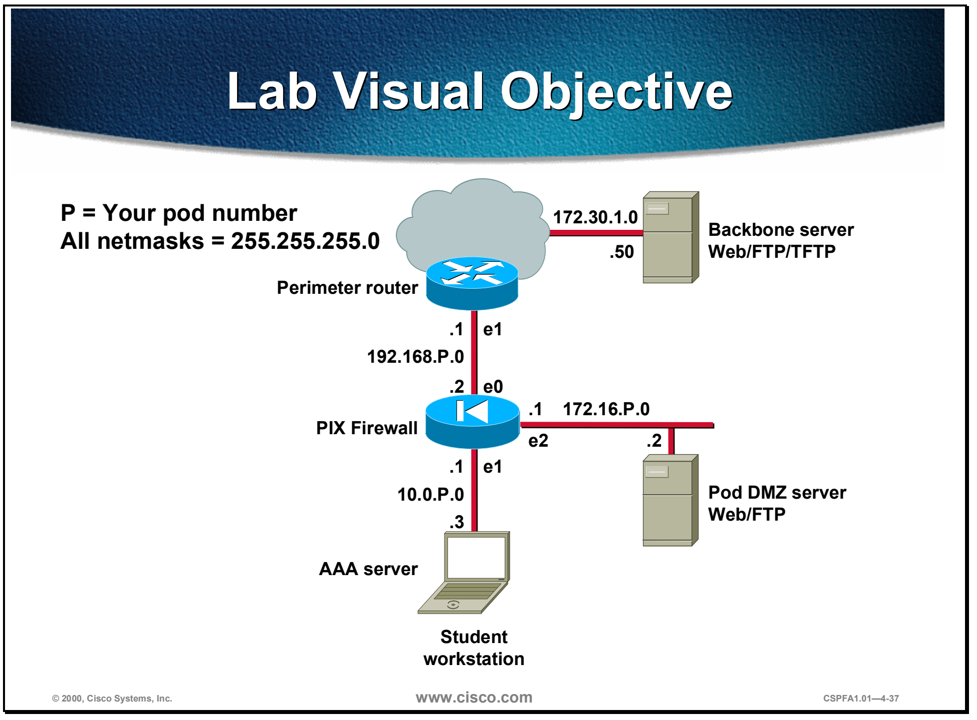
Objectives

In this lab exercise you will complete the following tasks:

- Install Cisco Secure ACS for Windows NT server.
- Add a user to the Cisco Secure ACS database.
- Identify a AAA server and protocol.
- Configure and test inbound authentication.
- Configure and test outbound authentication.
- Configure and test console access authentication.
- Configure and test Virtual Telnet authentication.
- Change and test authentication timeouts and prompts.
- Configure and test authorization.
- Configure and test accounting.

Visual Objective

The following figure displays the configuration you will complete in this lab exercise.



Task 1: Install Cisco Secure ACS for Windows NT Server

Perform the following steps to install Cisco Secure ACS on your Windows NT server:

Step 1 Install Cisco Secure ACS on your Windows NT server from the CD-ROM or from the files on your hard drive, as indicated by the instructor.

- When installing from the CD-ROM, complete the following:
  - Windows NT will automatically start the autorun.exe program and you are prompted to install Cisco Secure ACS.
  - Click **Install** to start the installation process.
- When installing from files in your hard drive, complete the following:
  - Open the folder where the installation files are located and double-click the setup.exe program to start installation.
  - Or choose **Start>Run…** and enter **setup.exe** with a full path to the file and start installation.

Step 2 Click **ACCEPT** to accept the Software License Agreement.

Step 3 Read the Welcome panel. Click **Next** to continue.

Step 4 Read and check all four items in the Before You Begin panel. This is a reminder of things you should do prior to installation. Click **Next** to continue.

Step 5 Use the default installation folder indicated in the Choose Destination Location panel. Click **Next** to continue.

Step 6 Verify **Check the Cisco Secure ACS database only** is already selected in the Authentication Database Configuration panel. Click **Next** to continue.

Step 7 Enter the following information in the Cisco Secure ACS Network Access Server Details panel:

- Authenticate users: **TACACS+ (Cisco)**
- Access server name: **pixP** (see note below)
- Access server IP address: **10.0.P.1**
- Windows NT Server IP address: **10.0.P.3**
- TACACS+ or RADIUS key: **secretkey**

(where *P* =pod number)

Step 8 Click **Next** to start the f ile installation process.

Step 9 Select all six items displayed in the Advanced Options panel. Click **Next** to continue.

Step 10 Verify that **Enable Log-in Monitoring** is already selected in the Active Service Monitoring panel. Click **Next** to continue.

CAUTION *Do* not *select "Yes, I want to configure Cisco IOS software now" in the "Network Access Server Configuration" panel; this only applies to Cisco IOS™ routers.*

Step 11 Click **Next** to continue.

Step 12 Verify that the following are already selected in the Cisco Secure ACS Service Initiation panel:

- Yes, I want to start the Cisco Secure ACS Service now
- Yes, I want Setup to launch the Cisco Secure ACS Administrator from my browser following installation

---

Note Do *not* select "Yes, I want to review the Readme file."

Step 13 Click **Next** to start the Cisco Secure ACS servi ce.

Step 14 Read the Setup Complete panel and then click **Finish** to end the installation wizard and start your web browser with Cisco Secure ACS.

Task 2: Add a User to the Cisco Secure ACS Database

Perform the following steps to add a user to the Cisco Secure ACS database in your Windows NT server:

Step 1 The Cisco Secure ACS interface should now be displayed in your web browser. Click **User Setup** to open the User Setup interface.

Step 2 Add a user by entering **aaauser** in the user field.

Step 3 Click **Add/Edit** to go into the user information edit window.

Step 4 Give the user a password by entering **aaapass** in both the Password and Confirm Password fields.

Step 5 Click **Submit** to add the new user to the Cisco Secure ACS database. Wait for the interface to return to the User Setup main window.

Task 3: Identify a AAA Server and Protocol

Perform the following steps to identify a AAA server and a AAA protocol on the PIX Firewall:

Step 1 Create a group tag called MYTACACS and assign the TACACS+ protocol to it:

```
pixfirewall(config)# aaa-server MYTACACS protocol tacacs+
```

Step 2 Assign the Cisco Secure ACS IP address and the encryption key secretkey.

```
pixfirewall(config)# aaa-server MYTACACS (inside) host 10.0.P.3 secretkey.
```

Step 3 Verify your configuration:

```
pixfirewall(config)# show aaa-server
aaa-server MYTACACS protocol tacacs+
aaa-server MYTACACS (inside) host 10.0.P.3 secretkey timeout 5(P=your pod number)
aaa-server RADIUS protocol radius
```

(where P =pod number, and Q =peer pod number)

Task 4: Configure and Test Inbound Authentication

Perform the following steps to enable the use of inbound authentication on the PIX Firewall:

Step 1 Configure the PIX Firewall to require authentication for all inbound traffic:

```
pixfirewall(config)# aaa authentication include any inbound 0.0.0.0 0.0.0.0
    0.0.0.0 0.0.0.0 TACACS+
```

Step 2 Verify your configuration:

```
pixfirewall(config)# show aaa authentication
aaa authentication include any inbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 TACACS+
```

Step 3 Enable console logging of all messages:

```
pixfirewall(config)# logging console debug
```

Note If your web browser is open, close it. Choose **File>Close** from the web browser's menu.

Step 4 You must now test a peer pod inbound web authentication. Open your web browser, and go to a peer's DMZ web server:

http://192.168.Q.11

(where *Q* = peer pod number)

Step 5 When the web browser prompts you, ener **aaauser** for the username and **aaapass** for the password. On your PIX Firewall console, you should see the following:

```
109001: Auth start for user '???' from 192.168.Q.10/1726 to 10.0.P.2/80
109011: Authen Session Start: user 'aaauser', sid 0
109005: Authentication succeeded for user 'aaauser' from 10.0.P.2/80 to
    192.168.Q.10/1921 on interface outside
302001: Built outbound TCP connection 3928 for faddr 192.168.Q.10/1921 gaddr
    192.168.P.10/80 laddr 10.0.P.3/80 (aaauser)
```

(where *P* = pod number, and *Q* = peer pod number)

Step 6 After a peer successfully authenticates to your PIX Firewall, display your PIX Firewall authentication statistics:

```
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users       1          1
Authen In Progress        0          1
user 'pixuser' at 192.168.Q.10, authenticated
   absolute   timeout: 0:05:00
   inactivity timeout: 0:00:00
```


Task 5: Configure and Test Outbound Authentication

Perform the following steps to enable the use of outbound authentication on the PIX Firewall:

Step 1 Configure the PIX Firewall to require authentication for all outbound traffic:

```
pixfirewall(config)# aaa authentication include any outbound 0.0.0.0 0.0.0.0
    0.0.0.0 0.0.0.0 MYTACACS
```

Step 2 Verify your configuration:

```
pixfirewall(config)# show aaa authentication
aaa authentication include any outbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
aaa authentication include any inbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
```

Step 3 Test FTP outbound authentication from your Windows NT server:

```
C:\> ftp 172.30.1.50
Connected to 172.30.1.50
220-FTP authentication :
220
User (172.30.1.50:(none)): aaauser@ftpuser
331-Password:
331
Password: aaapass@ftppass
230-220 172.30.1.50 FTP server ready.
331-Password required for ftpuser
230-User ftpuser logged in.
230
ftp>
```

On your PIX Firewall console, you should see the following:

```
109001: Auth start for user '???' from 10.0.P.3/1726 to 172.30.1.50/21
109011: Authen Session Start: user 'aaauser', sid 11
109005: Authentication succeeded for user 'aaauser' from 10.0.P.3/1726 to
    172.30.1.50/21 on interface inside
302001: Built outbound TCP connection 3928 for faddr 172.30.1.50/21 gaddr
    192.168.P.10/1726 laddr 10.0.P.3/1726 (aaauser)
```

(where *P* = pod number)

Step 4 Display authentication statistics on the PIX Firewall:

```
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users       1          1
Authen In Progress        0          1
user 'pixuser' at 10.0.P.2, authenticated (P = your pod number)
   absolute   timeout: 0:05:00
   inactivity timeout: 0:00:00
```

Step 5 Clear the uauth timer:

```
pixfirewall(config)# clear uauth
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users       0          1
```

```
Authen In Progress       0          1
```

Note If your web browser is open, close it. Choose **File>Exit** from the web browser's menu.

Step 6 Test web outbound authentication. Open your web browser and go to the following URL:

```
http://172.30.1.50
```

Step 7 When the web browser prompts you for a username and password, enter **aaauser**:

```
User Name: aaauser
Password: aaauser
```

Step 8 Display authentication statistics on the PIX Firewall:

```
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users      1          1
Authen In Progress       0          1
user 'pixuser' at 10.0.P.2, authenticated
   absolute   timeout: 0:05:00
   inactivity timeout: 0:00:00
```

(where *P* = pod number)

Task 6: Configure and Test Console Access Authentication

Perform the following steps to enable console Telnet authentication at the PIX Firewall:

Step 1 Configure the PIX Firewall to require authentication for Telnet console connections:

```
pixfirewall(config)# aaa authentication telnet console MYTACACS
```

Step 2 Verify your configuration:

```
pixfirewall(config)# show aaa authentication
aaa authentication include any outbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
aaa authentication include any inbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
aaa authentication include any any 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
```

Step 3 Configure the PIX Firewall to allow console Telnet logins:

```
pixfirewall(config)# telnet 10.0.P.1 255.255.255.0 inside
```

(where *P* = pod number)

Step 4 Verify your configuration:

```
pixfirewall(config)# show telnet
10.0.P.1 255.255.255.0 inside
```

(where *P* = pod number)

Step 5 Clear the uauth timer:

```
pixfirewall(config)# clear uauth
```



```
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users       0          1
Authen In Progress        0          1
```

Step 6 Telnet to the PIX Firewall console:

```
C:\> telnet 10.0.P.1
PIX passwd: cisco

Welcome to the PIX firewall

Copyright (c) 1996-1999 by Cisco Systems, Inc.

                Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

                Cisco Systems, Inc.
                170 West Tasman Drive
                San Jose, California 95134-1706

Username: aaauser
Password: aaapass
Type help or '?' for a list of available commands.
pixfirewall>
```

(where*P* = pod number)

On your PIX Firewall console, you should see the following:

```
307002: Permitted Telnet login session from 10.0.P.3
111006: Console Login from aaauser at console
```

(where *P* = pod number)

Task 7: Configure and Test Virtual Telnet Authentication

Perform the following steps to enable the use of authentication with virtual Telnet on the PIX Firewall:

Step 1 Configure the PIX Firewall to accept authentication to a virtual Telnet service:

```
pixfirewall(config)# virtual telnet 192.168.P.5
```

(where P=pod number, and Q=peer pod number)

Step 2 Verify the virtual Telnet configuration:

```
pixfirewall(config)# show virtual telnet
virtual telnet 192.168.P.5
```

(where *P* = pod number)

Step 3 Clear the uauth timer:

```
pixfirewall(config)# clear uauth
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users       0          1
Authen In Progress        0          1
```

Step 4 Telnet to the virtual Telnet IP address to authenticate from your Windows NT server:

```
C:\> telnet 192.168.P.5
LOGIN Authentication

Username: aaauser

Password: aaapass


Authentication Successful
```

(where *P* = pod number)

Note If your web browser is open, close it. Choose **File>Close** from the web browser's menu.

Step 5 Test that you are authenticated. Open your web browser and go to the following URL:

http://172.30.1.50

You should not be prompted to authenticate.

Step 6 Clear the uauth timer:

```
pixfirewall(config)# clear uauth
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users       0          1
Authen In Progress        0          1
```

Note If your web browser is open, close it. Choose **FileClose** from the web browser's menu.

Step 7 Test that you are *not* authenticated and need to reauthenticate. Open your web browser and go to the following URL:

http://172.30.1.50

Step 8 When the web browser prompts, enter **aaauser** for the username and **aaapass** for the password.

Task 8: Change and Test Authentication Timeouts and Prompts

Perform the following steps to change the authentication timeouts and prompts:

Step 1 View the current uauth timeout settings:

```
pixfirewall(config)# show timeout uauth
timeout uauth 0:05:00 absolute uauth 0:00:00 inactivity
```

Step 2 Set the uauth absolute timeout to 3 hours:

```
pixfirewall(config)# timeout uauth 3 absolute
```

Step 3 Set the uauth inactivity timeout to 30 minutes:

```
pixfirewall(config)# timeout uauth 0:30 inactivity
```

Step 4 Verify the new uauth timeout settings:

```
pixfirewall(config)# show timeout uauth
timeout uauth 3:00:00 absolute uauth 0:30:00 inactivity
```

Step 5 View the current authentication prompt settings:

```
pixfirewall(config)# show auth-prompt
```

Nothing should be displayed.

Step 6 Set the prompt that users get when authenticating:

```
pixfirewall(config)# auth-prompt prompt Please Authenticate to the Firewall
```

Step 7 Set the message that users get when successfully authenticating:

```
pixfirewall(config)# auth-prompt accept You’ve been Authenticated
```

Step 8 Set the message that users get when their authentication is rejected:

```
pixfirewall(config)# auth-prompt reject Authentication Failed, Try Again
```

Step 9 Verify the new prompt settings:

```
pixfirewall(config)# show auth-prompt
auth-prompt prompt Please Authenticate to the Firewall
auth-prompt accept You've been Authenticated
auth-prompt reject Authentication Failed, Try Again
```

Step 10 Clear the uauth timer:

```
pixfirewall(config)# clear uauth
pixfirewall(config)# show uauth
                        Current   Most Seen
Authenticated Users       0          1
Authen In Progress        0          1
```

Step 11 Telnet to the virtual Telnet IP address to test your new authentication prompts. From your Windows NT server, enter the following:

```
C:\> telnet 192.168.P.5

LOGIN Authentication

Please Authenticate to the Firewall
```

```
Username: wronguser

Password: wrongpass Authentication Failed, Try Again
LOGIN Authentication

Please Authenticate to the Firewall
Username: aaauser

Password: aaapass
You've been Authenticated

Authentication Successful
```

(where *P* = pod number)

Task 9: Configure and Test Authorization

Perform the following steps to enable the use of authorization on the PIX Firewall:

Step 1 Configure the PIX Firewall to require authorization for all outbound FTP traffic:

```
pixfirewall(config)# aaa authorization include ftp outbound 0.0.0.0 0.0.0.0
    0.0.0.0 0.0.0.0 MYTACACS
```

Step 2 Configure the PIX Firewall to require authorization for all outbound ICMP traffic:

```
pixfirewall(config)# aaa authorization include icmp/8 outbound 0.0.0.0 0.0.0.0
    0.0.0.0 0.0.0.0 MYTACACS
```

Step 3 Verify your configuration:

```
pixfirewall(config)# show aaa authorization
aaa authorization include ftp outbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
aaa authorization include 1/8 outbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
```

Step 4 Test ICMP Echo Request failure from your Windows NT server:

```
C:\> ping 172.30.1.50
Pinging 172.30.1.50 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

On your PIX Firewall console, you should see the following:

```
109001: Auth start for user 'aaauser' from 10.0.P.3/0 to 172.30.0.50/0
109008: Authorization denied for user 'aaauser' from 10.0.P.2/0 to 172.30.0.50/0
    on interface inside
```

(where *P* = pod number)

Step 5 Test FTP authorization failure from your Windows NT server:

```
C:\> ftp 172.30.1.50
Connected to 172.30.1.50
220-FTP authentication :
220
```



```
User (172.30.1.50:(none)): aaauser@ftpuser
331-Password:
331
Password: aaapass@ftppass
530
530—Authorization Denied
530

Error:  Connection closed by foreign host.
```

On your PIX Firewall console, you should see the following:

```
109001: Auth start for user '???' from 10.0.P.3/1364 to 172.30.1.50/21
109011: Authen Session Start: user 'aaauser', sid 5
109005: Authentication succeeded for user 'aaauser' from 10.0.P.3/1364 to
   172.30.1.50/21 on interface inside
109008: Authorization denied for user 'aaauser' from 10.0.P.3/1364 to
   172.30.1.50/21 on interface inside
```

(where *P* = pod number)

Step 6 Click **Group Setup** to open the Group Setup interface.

Step 7 Choose **default group (user)** from the Group pull-down menu.

Step 8 Verify that your user belongs to the selected group. Click **Users in Group** to display the users under that group. The following information should be shown for the user:

- User: **aaauser**
- Status: **Enabled**
- Group: **Default Group (1 user)**

Step 9 Click **Edit Settings** to go to the Group Settings for your group.

Step 10 Scroll down the Group Settings until you find IOS Commands. Select the **IOS Commands** checkbox.

Step 11 Check the **Command:** checkbox under IOS Commands.

Step 12 Enter **ftp** in the Command field.

Step 13 Enter **permit 172.30.1.50** in the Arguments field.

Step 14 Click **Submit** to save the changes. Wait for the interface to return to the Group Setup main window.

Step 15 Click **Edit Settings** to go to the Group Settings for your group again.

Step 16 Scroll down the Group Settings until you find IOS Commands.

Step 17 Check the new **Command:** checkbox.

Step 18 Enter **1/8** in the Command field.

Step 19 Select **Permit** in the Unlisted arguments field.

Step 20 Click **Submit + Restart** to save the changes and restart Cisco Secure ACS. Wait for the interface to return to the Group Setup main window.

Step 21 Test FTP authorization success from your Windows NT server:

```
C:\> ftp 172.30.1.50
Connected to 172.30.1.50
220-FTP authentication :
220
User (172.30.1.50:(none)): aaauser@ftpuser
331-Password:
331
Password: aaapass@ftppass
230-220 172.30.1.50 FTP server ready.
331-Password required for ftpuser
230-User ftpuser logged in.
230
ftp>
```

On your PIX Firewall console, you should see the following:

```
109001: Auth start for user '???' from 10.0.P.3/1726 to 172.30.1.50/21
109011: Authen Session Start: user 'aaauser', sid 11
109005: Authentication succeeded for user 'aaauser' from 10.0.P.3/1726 to
    172.30.1.50/21 on interface inside
109011: Authen Session Start: user 'aaauser', sid 11
109007: Authorization permitted for user 'aaauser' from 10.0.P.3/1726 to
    172.30.1.50/21 on interface inside
302001: Built outbound TCP connection 3928 for faddr 172.30.1.50/21 gaddr
    192.168.P.10/1726 laddr 10.0.P.3/1726 (aaauser)
```

(where *P* = pod number)

Step 22 Test ICMP Echo Request success from your Windows NT server:

```
C:\> ping 172.30.1.50
Pinging 172.30.1.50 with 32 bytes of data:

Request timed out.
Request timed out.
Reply from 172.30.1.50: bytes=32 time<10ms TTL=128
Reply from 172.30.1.50: bytes=32 time<10ms TTL=128
```

On your PIX Firewall console, you should see the following:

```
109001: Auth start for user 'aaauser' from 10.0.P.3/0 to 172.30.1.50/0
109011: Authen Session Start: user 'aaauser', sid 1
109007: Authorization permitted for user 'aaauser' from 10.0.P.2/0 to
    172.30.1.50/0 on interface inside
```

(where *P* = pod number)

Task 10: Configure and Test Accounting

Perform the following steps to enable the use of accounting on the PIX Firewall:

Step 1 Configure the PIX Firewall to do accounting for all outbound traffic:

```
pixfirewall(config)# aaa accounting include any outbound 0.0.0.0 0.0.0.0 0.0.0.0
    0.0.0.0 MYTACACS
```

Step 2 Verify your configuration:


```
pixfirewall(config)# show aaa accounting
aaa accounting include any outbound 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 MYTACACS
```

Step 3 Clear the uauth timer:

```
pixfirewall(config)# clear uauth
pixfirewall(config)# show uauth
                        Current    Most Seen
Authenticated Users       0          1
Authen In Progress        0          1
```

Step 4 Test FTP outbound accounting from your Windows NT server:

```
C:\> ftp 172.30.1.50
Connected to 172.30.1.50
220-FTP authentication :
220
User (172.30.1.50:(none)): aaauser@ftpuser
331-Password:
331
Password: aaapass@ftppass
230-220 172.30.1.50 FTP server ready.
331-Password required for ftpuser
230-User ftpuser logged in.
230
ftp>
```

Step 5 View the accounting records. On Cisco Secure ACS, click **Reports and Activity** to open the Reports and Activity interface.

Step 6 Click the **TACACS+ Accounting** link.

Step 7 Click the **TACACS+ Accounting active.csv** link to open the accounting records. You should see the following:

Date	Time	User-Name	Group-Name	Caller-Id	Acct-Flags	•••	NAS-Portname	NAS-IP-Address	cmd
4/27/00	11:14:45	aaauser	Default Group	10.0.P.2	start	•••	PIX	10.0.P.1	ftp

Note If your web browser is open, close it. Choose **File>Exit** from the web browser's menu.

Step 8 Test web outbound accounting. Open your web browser and go to the following URL:

http://172.30.1.50

Step 9 When the web browser prompts you, enter **aaauser** for the username and **aaapass** for the password.

Step 10 Click the **TACACS+ Accounting** link.

Step 11 Click the **TACACS+ Accounting active.csv** link to open the accounting records. You should see the following:

Date	Time	User-Name	Group-Name	Caller-Id	Acct-Flags	• • •	NAS-Portname	NAS-IP Address	cmd
4/27/00	11:16:35	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:16:35	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
4/27/00	11:16:34	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:16:34	aaauser	Default Group	10.0.0.2	stop	• • •	PIX	10.0.0.1	http
4/27/00	11:16:34	aaauser	Default Group	10.0.0.2	stop	• • •	PIX	10.0.0.1	http
4/27/00	11:16:34	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:16:34	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:16:34	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:16:33	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:16:32	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:16:29	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	http
4/27/00	11:14:45	aaauser	Default Group	10.0.0.2	start	• • •	PIX	10.0.0.1	ftp

Step 12 Disable AAA by entering the following command.

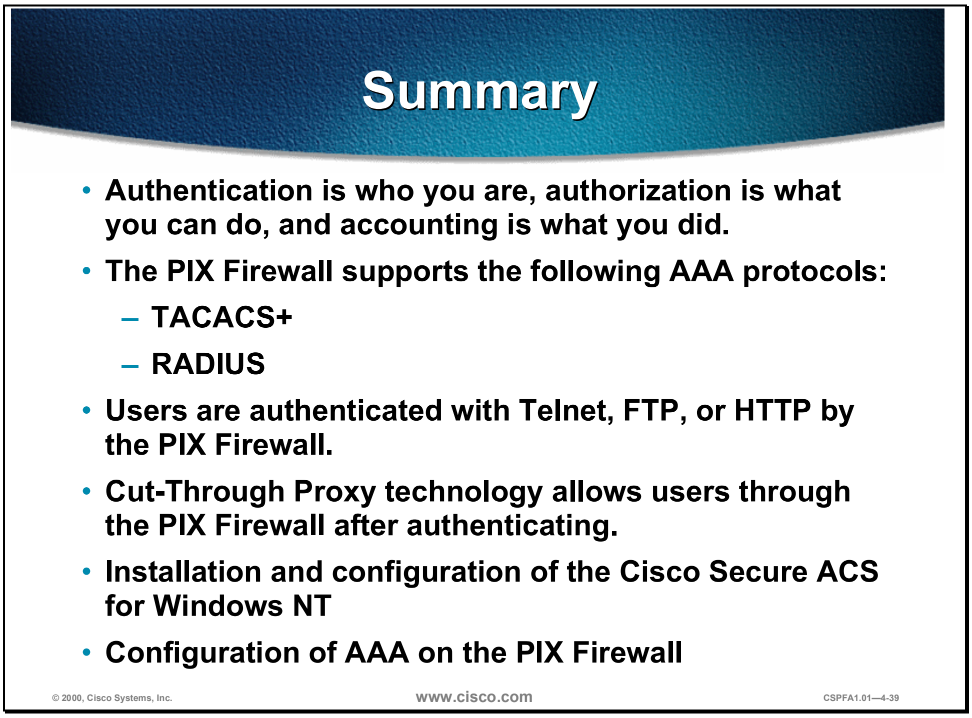
```
pixfirewall(config)# clear aaa
```

Step 13 Turn of the logging:

```
pixfirewall(config)# no logging console debug
```


Summary

This section summarizes what you have learned in this chapter.



5

Cisco Secure PIX Firewall Advanced Protocol Handling and Attack Guards

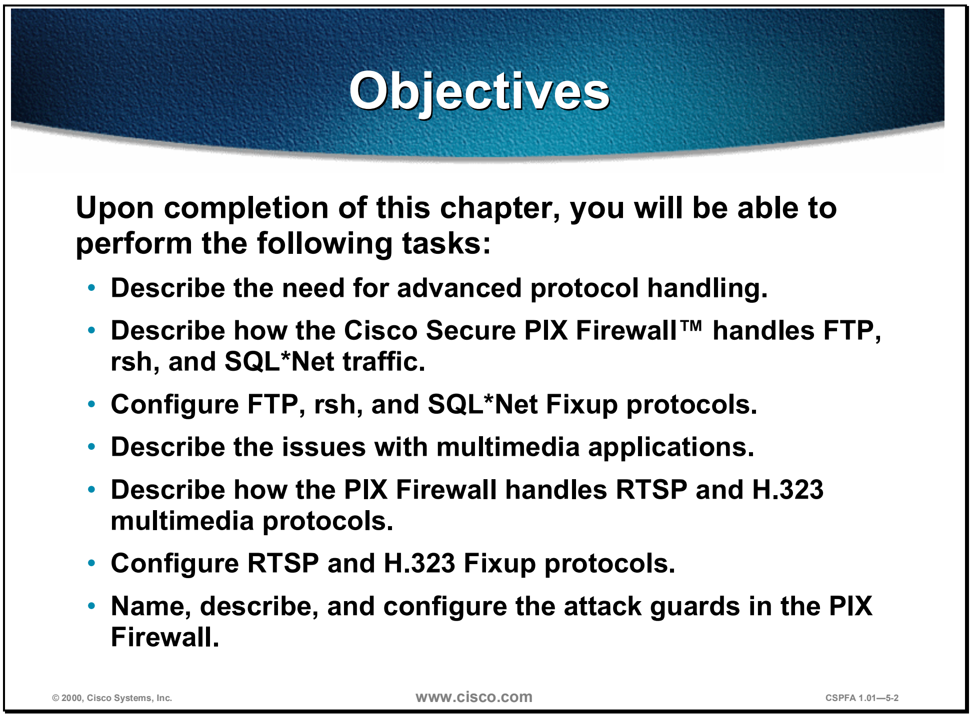
Overview

This chapter includes the following topics:

- Objectives
- Advanced protocols
- Multimedia support
- Security guards
- Lab exercise
- Summary

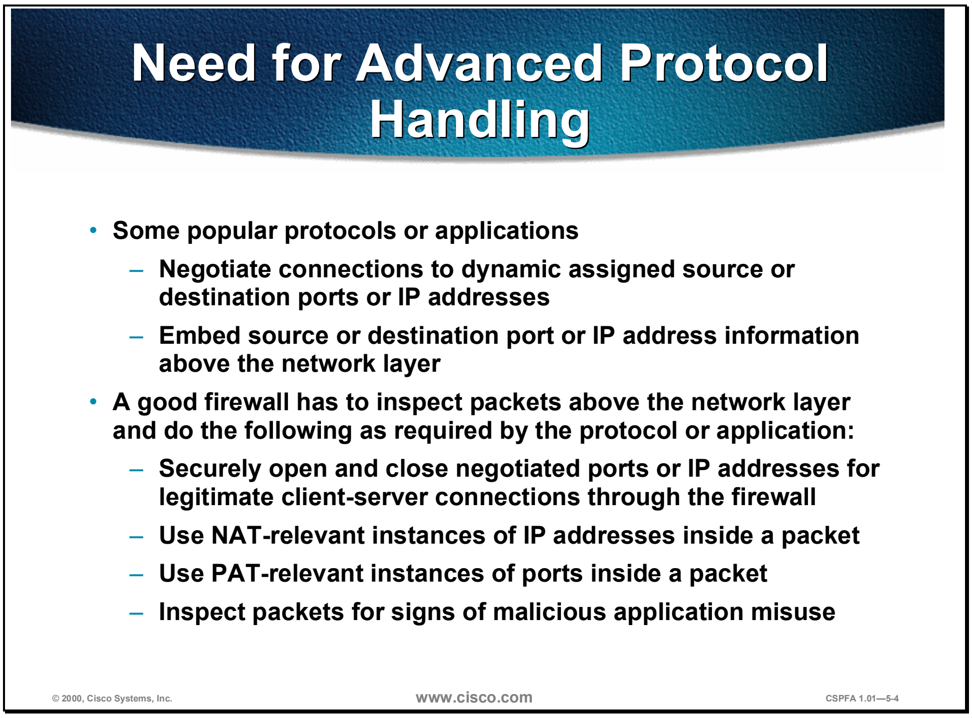
Objectives

This section lists the chapter's objectives.



Advanced Protocols

This section discusses the configuration and handling of the FTP, remote shell (rsh), and SQL protocols.

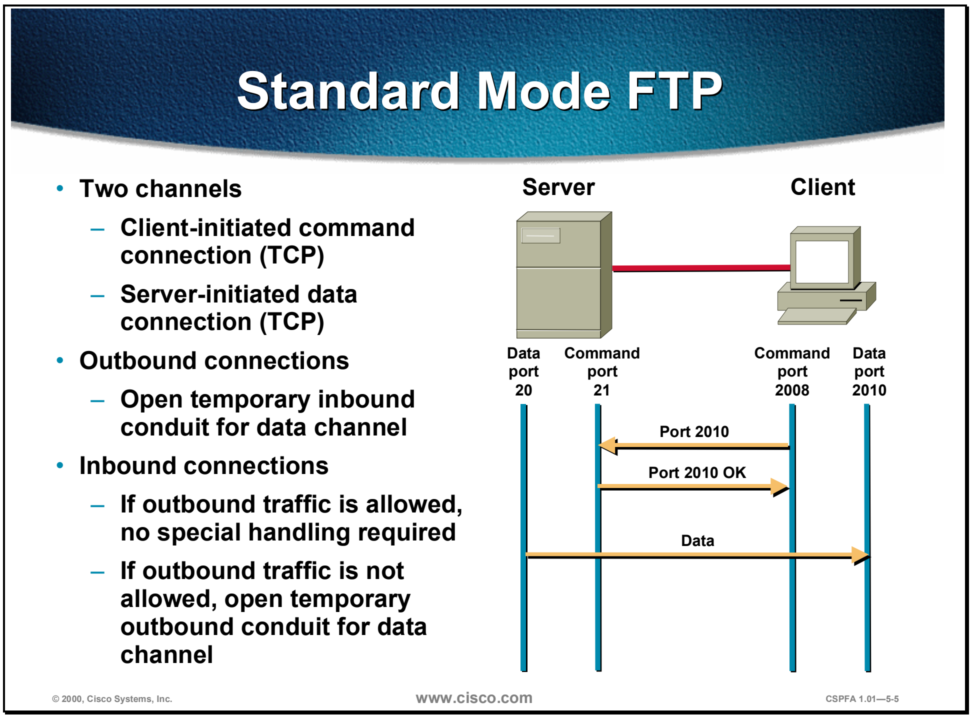


Today many corporations use the Internet for business transactions. For the corporations to keep their internal networks secure from potential threats from the Internet, they can implement firewalls on their internal network. Even though these firewalls help protect a corporation's internal networks from external threats, firewalls have caused problems as well: Some of the protocols and applications that the corporations use to communicate are not allowed through the firewalls. Specifically, protocols need to negotiate FTP, HTTP, H.323, and SQL*Net connections to dynamically assigned source or destination ports, or IP addresses, through the firewall.

A good firewall has to inspect packets above the network layer and do the following as required by the protocol or application:

- Securely open and close negotiated ports or IP addresses for legitimate client-server connections through the firewall
- Use Network Address Translation (NAT)-relevant instances of IP address inside a packet
- Use Port Address Translation (PAT)-relevant instances of ports inside a packet
- Inspect packets for signs of malicious application misuse

You can configure the Cisco Secure PIX™ Firewall to allow the required protocols or applications through. This enables a corporation's internal networks to remain secure while still being able to continue day-to-day business over the Internet.

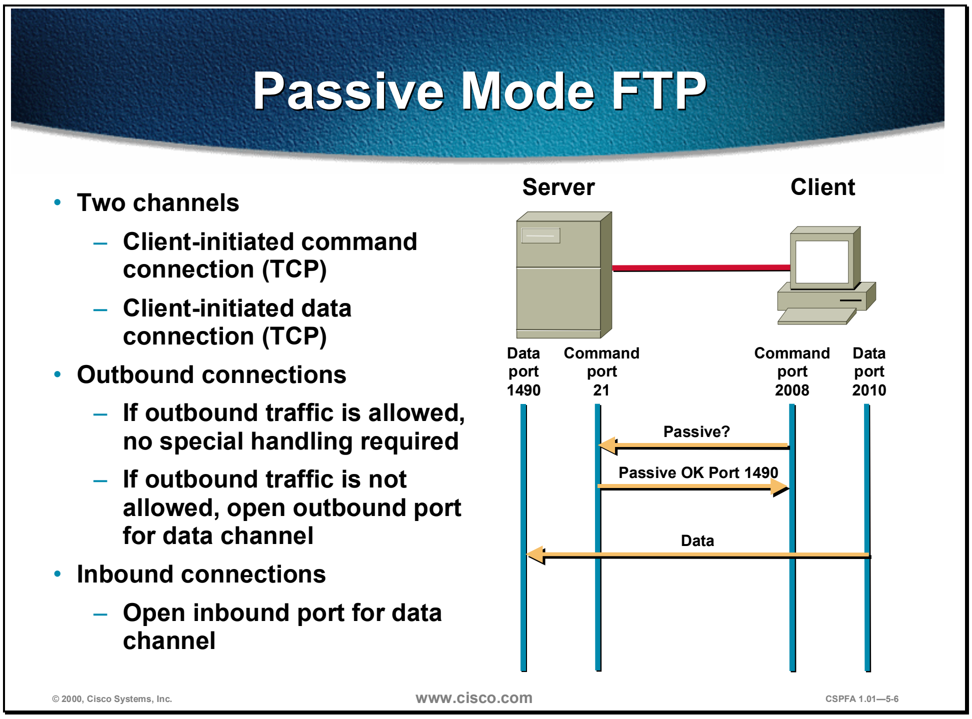


Standard mode FTP uses two channels for communications. When a client first starts an FTP connection, it opens a standard TCP channel from one of its high-order ports to port 21 on the server. This is referred to as the command channel. When the client requests data from the server, it tells the server to send the data to a given high-order port. The server acknowledges the request and initiates a connection from its own port 20 to the high-order port that the client requested. This is referred to as the data channel.

Because the server initiates the connection to the requested port on the client, it was difficult in the past to have firewalls allow this data channel to the client without permanently opening port 20 connections from outside servers to inside clients for outbound FTP connections. This created a potential vulnerability by exposing clients on the inside of the firewall.

For FTP traffic, the PIX Firewall behaves in the following manner:

- Outbound connections
  - When the client requests data, the PIX Firewall opens a temporary inbound conduit for the data channel from the server. This conduit is torn down after the data is sent.
- Inbound connections
  - If a conduit exists allowing inbound connections to an FTP server, and if all outbound TCP traffic is implicitly allowed, no special handling is required because the server initiates the data channel from the inside.
  - If a conduit exists allowing inbound connections to an FTP server, and if all outbound TCP traffic is not implicitly allowed, the PIX Firewall opens a temporary conduit for the data channel from the server. This conduit is torn down after the data is sent.

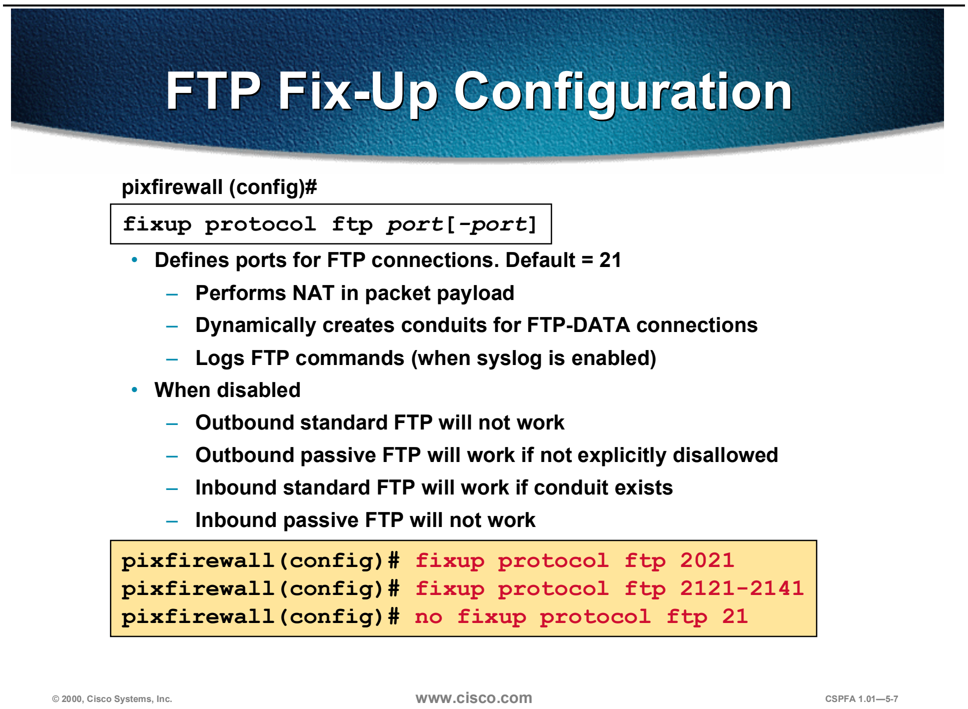


Passive mode PFTP also uses two channels for communications. The command channel works the same as in a standard FTP connection, but the data channel setup works differently. When the client requests data from the server, it asks the server if it accepts PFTP connections. If the server accepts PFTP connections, it sends the client a high-order port number to use for the data channel. The client then initiates the data connection from its own high-order port to the port that the server sent.

Because the client initiates both the command and data connections, early firewalls could easily support this without exposing inside clients to attack.

For PFTP traffic, the PIX Firewall behaves in the following manner:

- Outbound connections
  - If all outbound TCP traffic is implicitly allowed, no special handling is required because the client initiates both the command and data channels from the inside.
  - If all outbound TCP traffic is not implicitly allowed, the PIX Firewall opens a temporary conduit for the data channel from the client. This conduit is torn down after the data is sent.
- Inbound connections
  - If a conduit exists allowing inbound connections to a PFTP server, when data is requested by the client, the PIX Firewall opens a temporary inbound conduit for the data channel initiated by client. This conduit is torn down after the data is sent.



By default, the PIX Firewall inspects port 21 connections for FTP traffic. If you have FTP servers using ports other than port 21, you need to use the **fixup protocol ftp** command to have the PIX Firewall inspect these other ports for FTP traffic.

The **fixup protocol ftp** command causes the PIX Firewall to do the following for FTP traffic on the indicated port:

- Perform NAT or PAT in packet payload
- Dynamically create conduits for FTP data connections
- Log FTP commands (when Syslog is enabled)

Use the **no** form of the command to disable the inspection of traffic on the indicated port for FTP connections. If the **fixup protocol ftp** command is not enabled for a given port, then:

- Outbound standard FTP will *not* work properly on that port
- Outbound passive FTP will work properly on that port as long as outbound traffic is not explicitly disallowed
- Inbound standard FTP will work properly on that port if a conduit to the inside server exists
- Inbound passive FTP will *not* work properly on that port

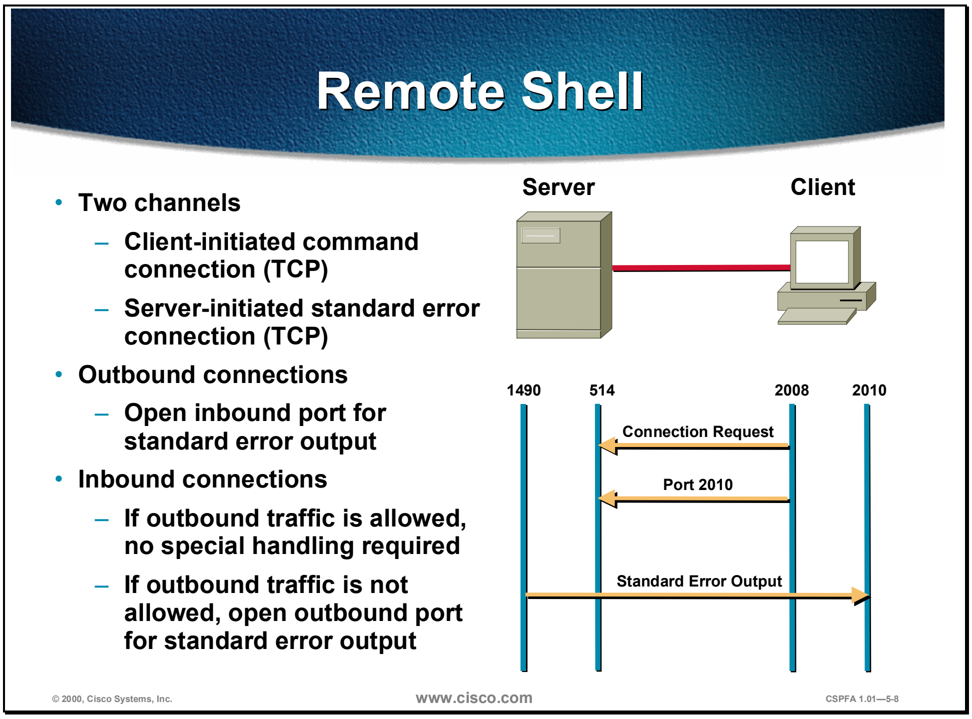
Using the **no fixup protocol ftp** command without any arguments causes the PIX Firewall to clear all previous **fixup protocol ftp** assignments and set port 21 back as the default.

The syntax of the **fixup protocol ftp** command is as follows:

fixup protocol ftp ***port*****[-*****port*****]**

no fixup protocol ftp ***port*****[-*****port*****]**

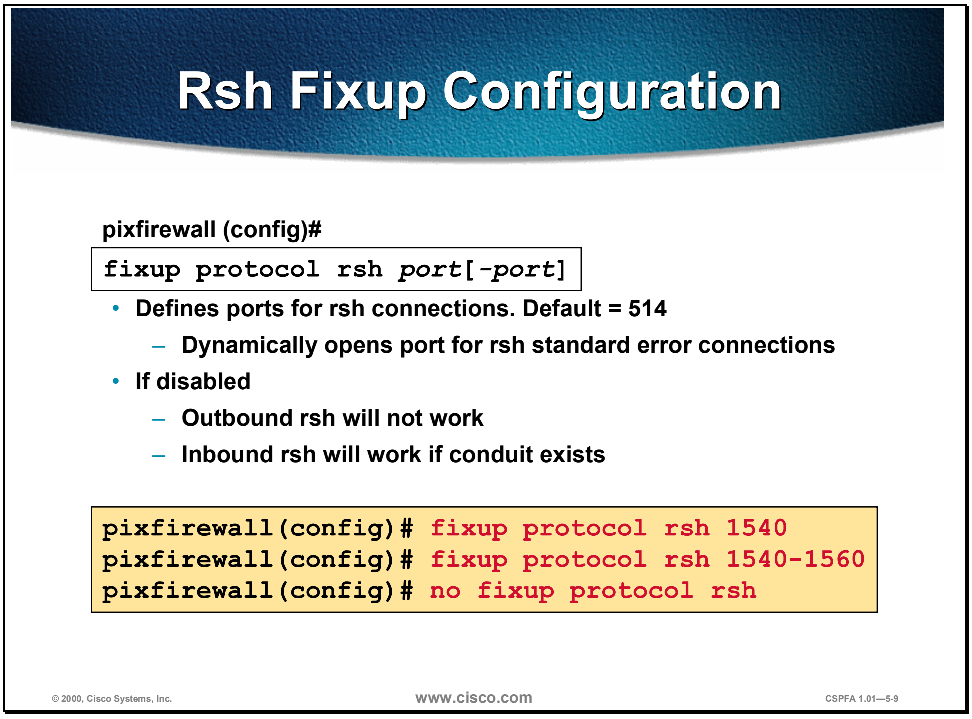
Argument	Description
***port*[-*port*]**	Single port or port range that the PIX Firewall will inspect for FTP connections.



Remote shell (rsh) uses two channels for communications. When a client first starts an rsh connection, it opens a standard TCP channel from one of its high-order ports to port 514 on the server. The server opens another channel for standard error output to the client.

For rsh traffic, the PIX Firewall behaves in the following manner:

- Outbound connections
  - When standard error messages are sent from the server, the PIX Firewall opens a temporary inbound conduit for this channel. This conduit is torn down when no longer needed.
- Inbound connections
  - If a conduit exists allowing inbound connections to an rsh server, and if all outbound TCP traffic is implicitly allowed, no special handling is required because the server initiates the standard error channel from the inside.
  - If a conduit exists allowing inbound connections to an rsh server, and if all outbound TCP traffic is *not* implicitly allowed, the PIX Firewall opens a temporary conduit for the standard error channel from the server. This conduit is torn down after the messages are sent.



By default, the PIX Firewall inspects port 514 connections for Rsh traffic. If you have Rsh servers using ports other than port 514, you need to use the **fixup protocol rsh** command to have the PIX Firewall inspect these other ports for Rsh traffic.

The **fixup protocol rsh** command causes the PIX Firewall to dynamically create conduits for Rsh standard error connections for Rsh traffic on the indicated port.

Use the **no** form of the command to disable the inspection of traffic on the indicated port for Rsh connections. If the **fixup protocol rsh** command is not enabled for a given port then

- Outbound Rsh will *not* work properly on that port
- Inbound Rsh will work properly on that port if a conduit to the inside server exists

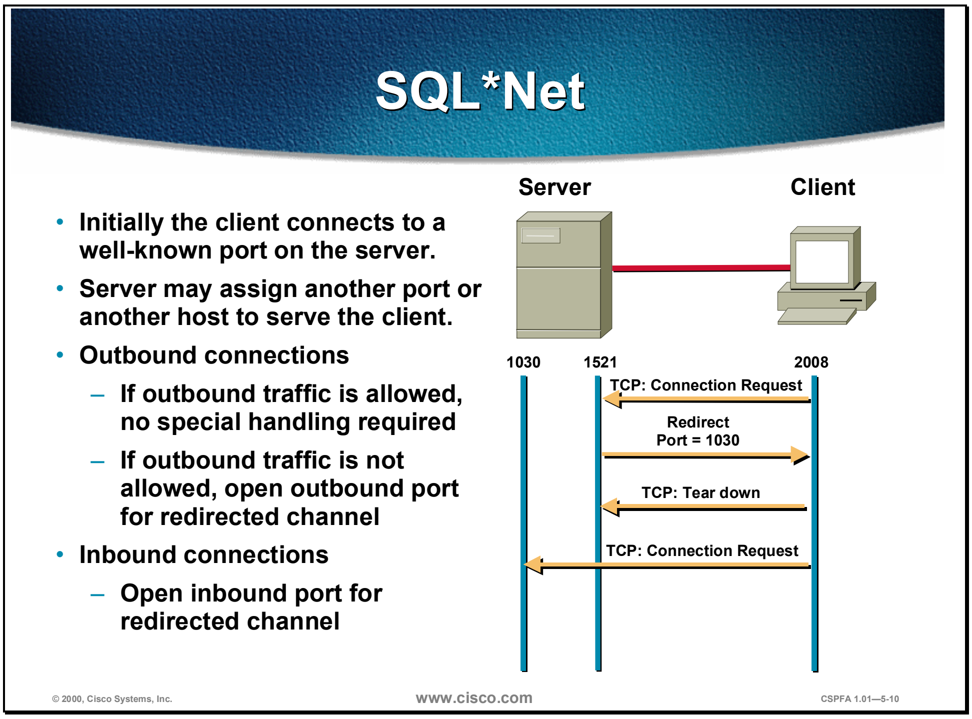
Using the **no fixup protocol rsh** command without any arguments causes the PIX Firewall to clear all previous **fixup protocol rsh** assignments and set port 514 back as the default.

The syntax of the **fixup protocol rsh** command is as follows:

fixup protocol rsh ***port*[-*port*]**

no fixup protocol rsh ***port*[-*port*]**

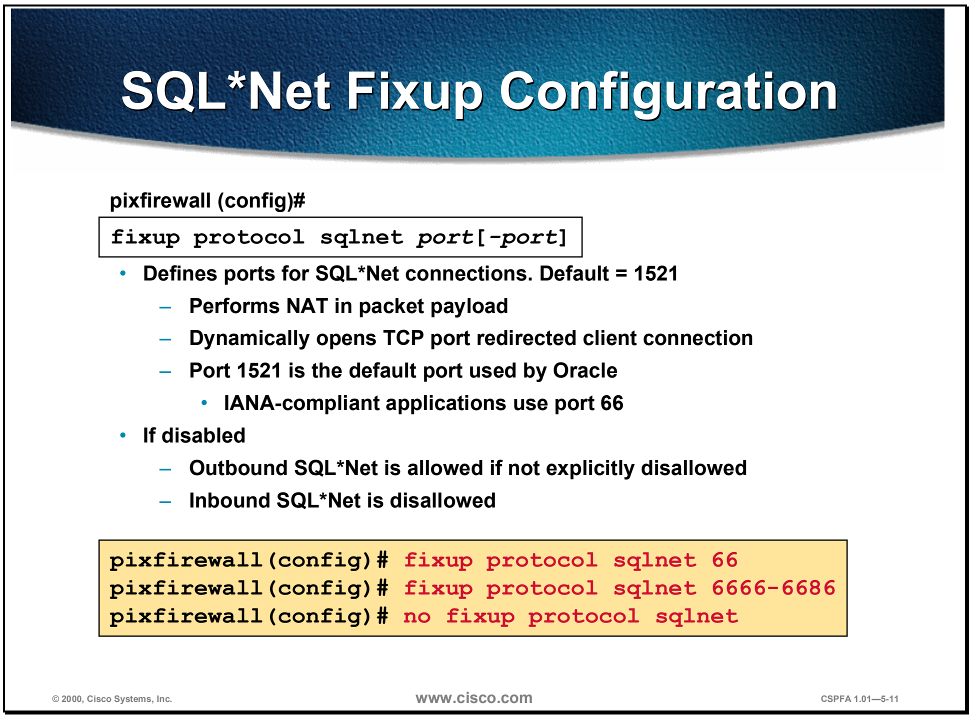
Argument	Description
***port*[-*port*]**	Single port or port range that the PIX Firewall will inspect for Rsh connections.



SQL*Net only uses one channel for communications but it could be redirected to a different port, and even more commonly to a different secondary server altogether. When a client first starts an SQL*Net connection, it opens a standard TCP channel from one of its high-order ports to port 1521 on the server. The server then proceeds to redirect the client to a different port or IP address. The client tears down the initial connection and establishes the second connection.

For SQL*Net traffic, the PIX Firewall behaves in the following manner:

- Outbound connections
  - If all outbound TCP traffic is implicitly allowed, no special handling is required because the client initiates all TCP connections from the inside.
  - If all outbound TCP traffic is *not* implicitly allowed, the PIX Firewall opens a conduit for the redirected channel between the server and the client.
- Inbound connections
  - If a conduit exists allowing inbound connections to an SQL*Net server, the PIX Firewall opens an inbound conduit for the redirected channel.



By default, the PIX Firewall inspects port 1521 connections for SQL*Net traffic. If you have SQL*Net servers using ports other than port 1521, you must use the **fixup protocol sqlnet** command to have the PIX Firewall inspect these other ports for SQL*Net traffic.

The **fixup protocol sqlnet** command causes the PIX Firewall to do the following for SQL*Net traffic on the indicated port:

- Perform NAT in packet payload
- Dynamically create conduits for SQL*Net redirected connections

Use the **no** form of the command to disable the inspection of traffic on the indicated port for SQL*Net connections. If the **fixup protocol sqlnet** command is not enabled for a given port, then

- Outbound SQL*Net will work properly on that port as long as outbound traffic is not explicitly disallowed
- Inbound passive SQL*Net will *not* work properly on that port

Using the **no fixup protocol sqlnet** command without any arguments causes the PIX Firewall to clear all previous **fixup protocol sqlnet** assignments and set port 1521 back as the default.

The syntax of the **fixup protocol sqlnet** command is as follows:

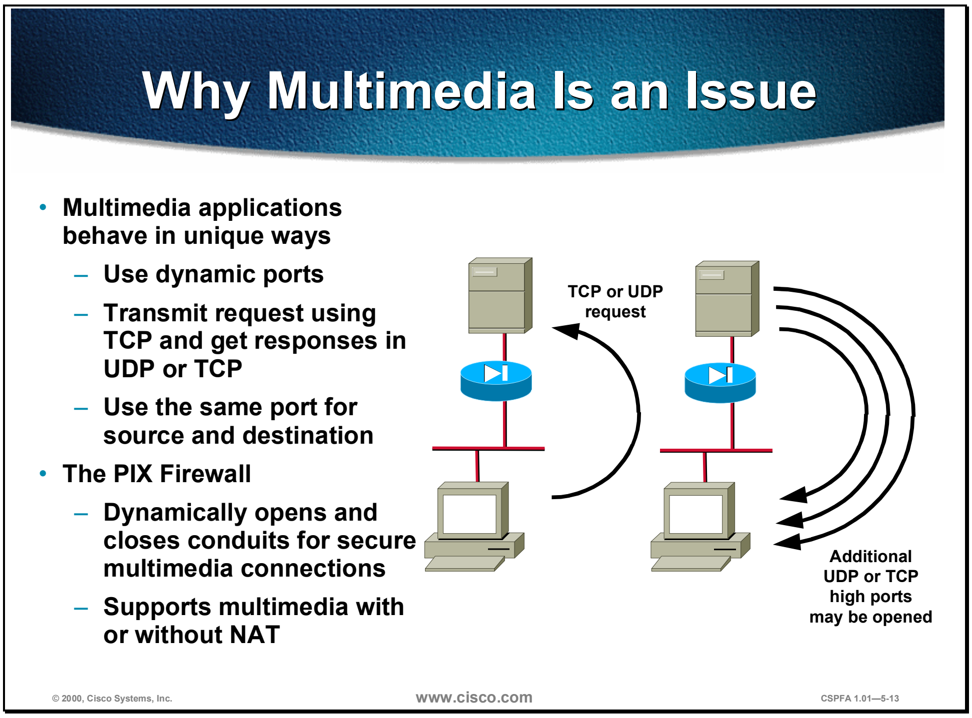
fixup protocol sqlnet ***port*****[-*****port*****]**

no fixup protocol sqlnet ***port*****[-*****port*****]**

Argument	Description
port**[-*****port*****]**	Single port or port range that the PIX Firewall will inspect for SQL*Net connections.

Multimedia Support

This section discusses multimedia: advantages and application supports, H.323 support, and important multimedia configurations.

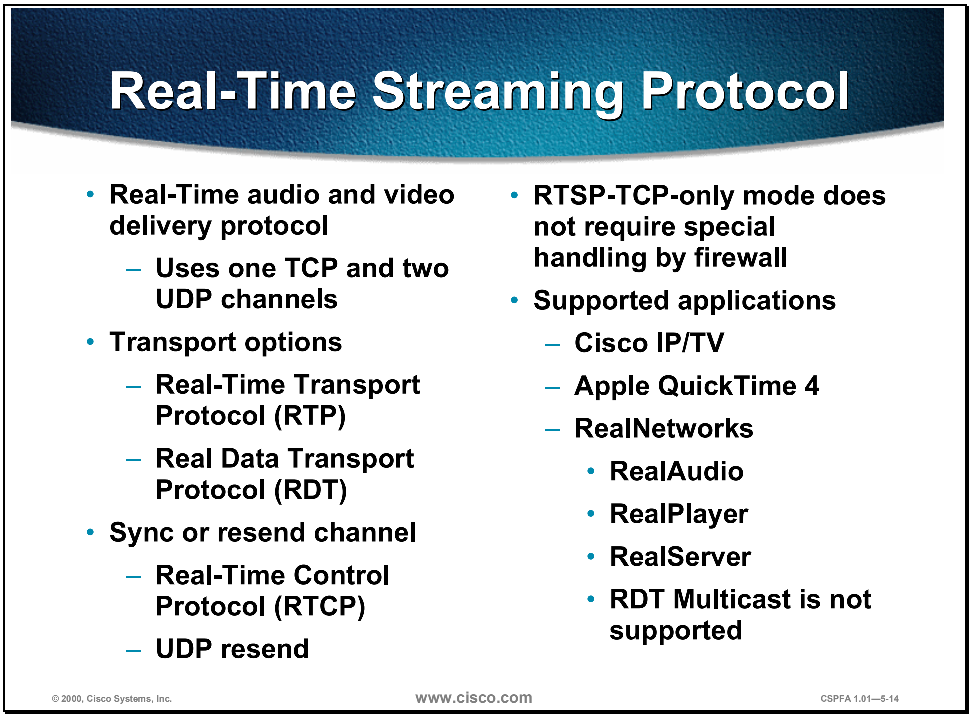


Multimedia applications may transmit requests on TCP, get responses on UDP or TCP, use dynamic ports, may use the same port for source and destination, and so on. Every application behaves in a different way. Implementing support for all multimedia applications using a single secure method is very difficult. Two examples of multimedia applications are given below:

- RealAudio sends the originating request to TCP port 7070. The RealAudio server replies with multiple UDP streams anywhere from UDP port 6970 through 7170 on the client machine.
- The CUseeMe client sends the originating request from TCP port 7649 to TCP port 7648. The CUseeMe datagram is unique in that it includes the legitimate IP address in the header as well as in the payload, and sends responses from UDP port 7648 to UDP port 7648.

The PIX Firewall dynamically opens and closes UDP ports for secure multimedia connections. You do not need to open a large range of ports, which creates a security risk, or have to reconfigure any application clients.

Also, the PIX Firewall supports multimedia with or without NAT. Many firewalls that cannot support multimedia with NAT limit multimedia usage to only registered users, or require exposure of inside IP addresses to the Internet. Lack of support for multimedia with NAT often forces multimedia vendors to join proprietary alliances with firewall vendors to accomplish compatibility for their applications.



The Real-Time Streaming Protocol (RTSP) is a real-time audio and video delivery protocol used by many popular multimedia applications. It uses one TCP channel and some times two additional UDP channels. RTSP applications use the well-known port 554, usually TCP and rarely UDP. RFC 2326 requires only TCP so the PIX Firewall only supports TCP. This TCP channel is the control channel and is used to negotiate the other two UDP cha,nnels depending on the transport mode that is configured on the client.

The first UDP channel is the data connection and may use one of the following transport modes:

- Real-Time Transport Protocol (RTP)
- Real Data Transport Protocol (RDT)

The second UDP channel is another control channel and it may use one of the following modes:

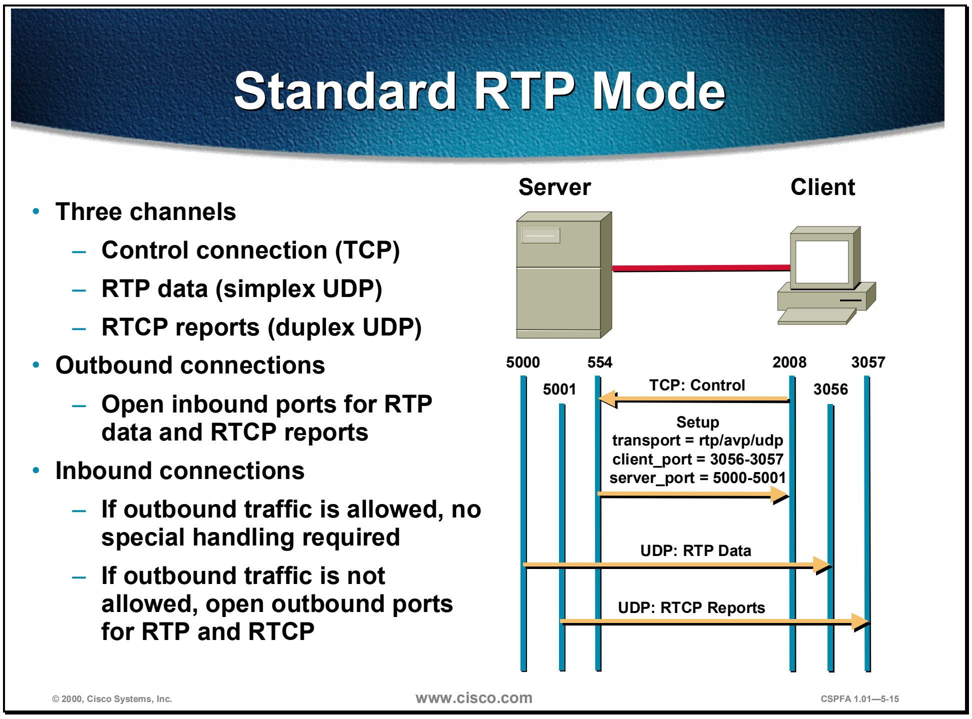
- Real-Time Control Protocol (RTCP)
- UDP Resend

RTSP supports a TCP-only mode. This mode contains only one TCP connection, which is used as the control and data channels. Because this mode contains only one constant standard TCP connection, no special handling by the PIX Firewall is required.

The following are RTSP applications supported by the PIX Firewall:

- Cisco IP/TV
- Apple QuickTime 4
- RealNetworks
  - RealAudio
  - RealPlayer
  - RealServer

Note RDT Multicast is not supported.

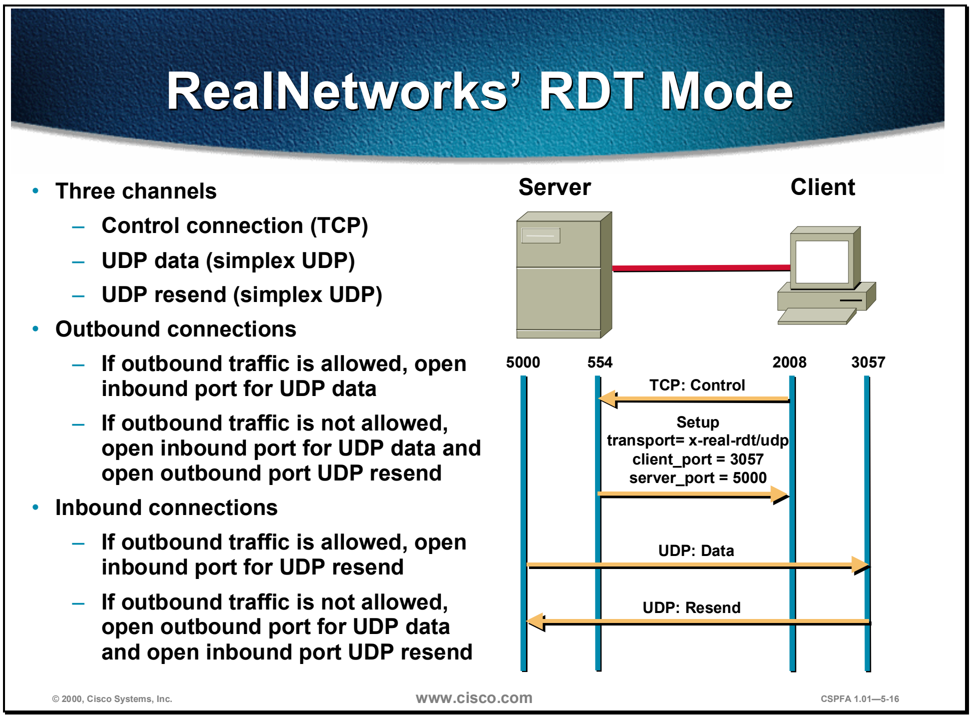


In standard RTP mode, the following three channels are used by RTSP:

- TCP control channel—Standard TCP connection initiated from the client to the server.
- RTP data channel—Simplex (unidirectional) UDP session used for media delivery using the RTP packet format from the sever to the client. The client's port is always an even numbered port.
- RTCP reports—Duplex (bidirectional) UDP session used to provide synchronization information to the client and packet loss information to the server. The RTCP port is always the next consecutive port from the RTP data port.

For standard RTP mode RTSP traffic, the PIX Firewall behaves in the following manner:

- Outbound connections
  - After the client and the server negotiate the transport mode and the ports to use for the sessions, the PIX Firewall opens temporary inbound conduits for the RTP data channel and RTCP report channel from the server.
- Inbound connections
  - If a conduit exists allowing inbound connections to an RTSP server, and if all outbound UDP traffic is implicitly allowed, no special handling is required since the server initiates the data and report channel from the inside.
  - If a conduit exists allowing inbound connections to an RTSP server, and if all outbound TCP traffic is *not* implicitly allowed, the PIX Firewall opens temporary conduits for the data and report channels from the server.



In RealNetworks' RDT mode, the following three channels are used by RTSP:

- TCP control channel—Standard TCP connection initiated from the client to the server.
- UDP data channel—Simplex (unidirectional) UDP session used for media delivery using the standard UDP packet format from the server to the client.
- UDP resend—Simplex (unidirectional) UDP session used for the client to request that the server resend lost data packets.

For RealNetworks' RDT mode RTSP traffic, the PIX Firewall will behave in the following manner:

- Outbound connections
  - If outbound UDP traffic is implicitly allowed, and after the client and the server negotiate the transport mode and the ports to use for the session, the PIX Firewall opens a temporary inbound conduit for the UDP data channel from the server.
  - If outbound UDP traffic is *not* implicitly allowed, and after the client and the server negotiate the transport mode and the ports to use for the session, the PIX Firewall opens a temporary inbound conduit for the UDP data channel from the server and a temporary outbound conduit for the UDP resend channel from the client.

- Inbound connections
  - If a conduit exists allowing inbound connections to an RTSP server, and if all outbound UDP traffic is implicitly allowed, the PIX Firewall opens a temporary outbound conduit for the UDP data channel from the server.
  - If a conduit exists allowing inbound connections to an RTSP server, and if all outbound TCP traffic is *not* implicitly allowed, the PIX Firewall opens temporary conduits for the UDP data and UDP resend channels from the server and client, respectively.

RTSPFixup Configuration

pixfirewall (config)#

```
fixup protocol rtsp port[-port]
```

- Defines ports for RTSP connections
  - No RTSP fixup is enabled by default. RFC2326 port is 554.
  - RTSP dynamically opens UDP connections as required by the RTSP transport.
  - PAT and dual NAT are not currently supported.
- If disabled
  - UDP transport modes are disallowed.
  - TCP transport modes are allowed (TCP connection rules apply).

```
pixfirewall(config)# fixup protocol rtsp 554
pixfirewall(config)# fixup protocol rtsp 8554-8574
pixfirewall(config)# no fixup protocol rtsp
```

 www.cisco.com CSPFA 1.01—5-17

By default, the PIX Firewall does not inspect any ports for RTSP connections. To enable the PIX Firewall to inspect specific ports for RTSP traffic, such as the standard port 554, use the **fixup protocol rtsp** command.

The **fixup protocol rtsp** command causes the PIX Firewall to dynamically create conduits for RTSP UDP channels for RTSP traffic on the indicated port.

Use the **no** form of the command to disable the inspection of traffic on the indicated port for RTSP connections. If the **fixup protocol sqlnet** command is not enabled for a given port, then neither outbound or inbound RTSP will work properly on that port.

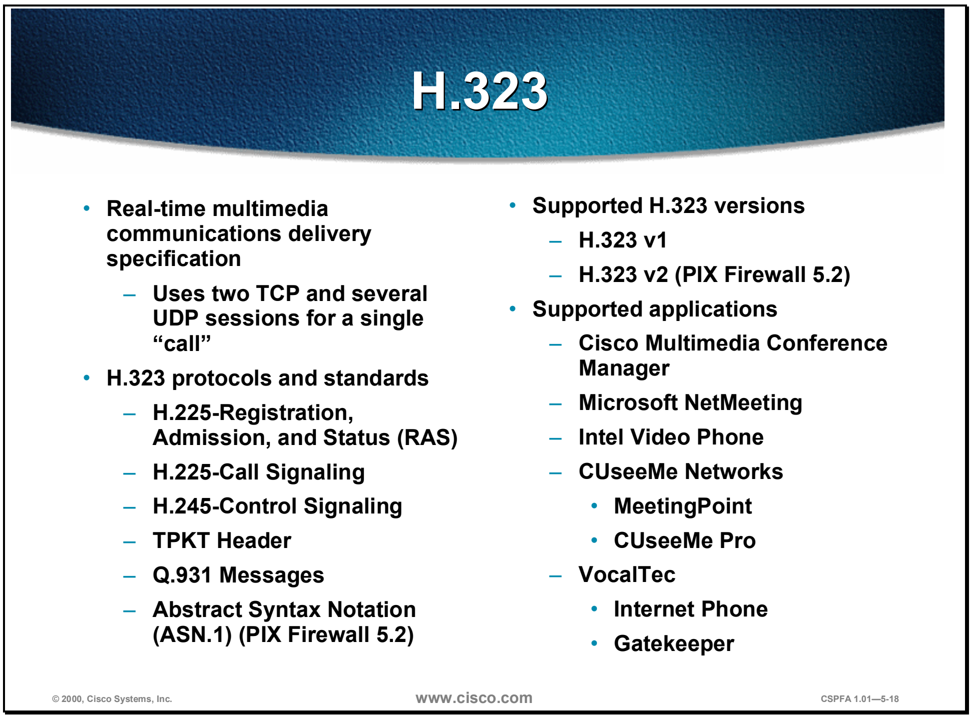
Using the **no fixup protocol rtsp** command without any arguments causes the PIX Firewall to clear all previous **fixup protocol rtsp** assignments.

The syntax of the **fixup protocol rtsp** command is as follows:

fixup protocol rtsp ***port*****[-*****port*****]**

no fixup protocol rtsp ***port*****[-*****port*****]**

Argument	Description
***port*[-*port*]**	Single port or port range that the PIX Firewall will inspect for RTSP connections.



H.323 is more complicated than other traditional protocols because it uses two TCP connections and several UDP sessions for a single "call." (Only one of the TCP connections goes to a well-known port; all the other ports are negotiated and are temporary.) Furthermore, the content of the streams is far more difficult for firewalls to understand than existing protocols because H.323 encodes packets using Abstract Syntax Notation, or ASN.1.

Other protocols and standards supported within H.323 are as follows:

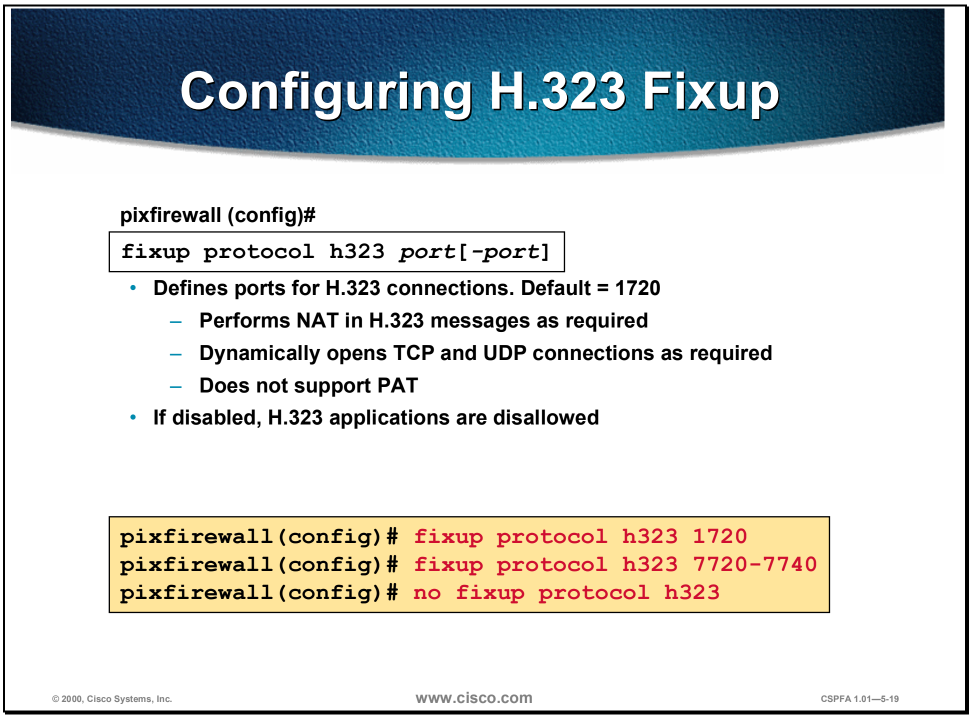
- H.225-Registration, Admission, and Status (RAS)
- H.225-Call Signaling
- H.245-Control Signaling
- TPKT Header
- Q.931 Messages
- Abstract Syntax Notation (ASN.1) (PIX Firewall 5.2)

Supported H.323 versions are as follows:

- H.323 v1
- H.323 v2 (PIX Firewall 5.2)

Supported applications are as follows:

- Cisco Multimedia Conference Manager
- Microsoft NetMeeting
- Intel Video Phone
- CUseeMe Networks
  - MeetingPoint
  - CUseeMe Pro
- VocalTec
  - Internet Phone
  - Gatekeeper



By default, the PIX Firewall inspects port 1720 connections for H.323 traffic. If you have H.323 servers using ports other than port 1720, you must use the **fixup protocol h323** command to have the PIX Firewall inspect these other ports for H.323 traffic.

The **fixup protocol h323** command causes the PIX Firewall to do the following for H.323 traffic on the indicated port:

- Perform NAT in packet payload
- Dynamically create conduits for TCP or UDP channels

Use the **no** form of the command to disable the inspection of traffic on the indicated port for H.323 connections. If the **fixup protocol h323** command is not enabled for a given port, then neither outbound nor inbound H.323 will work properly on that port.

Using the **no fixup protocol h323** command without any arguments causes the PIX Firewall to clear all previous **fixup protocol h323** assignments and set port 1720 back as the default.

The syntax of the **fixup protocol h323** command is as follows:

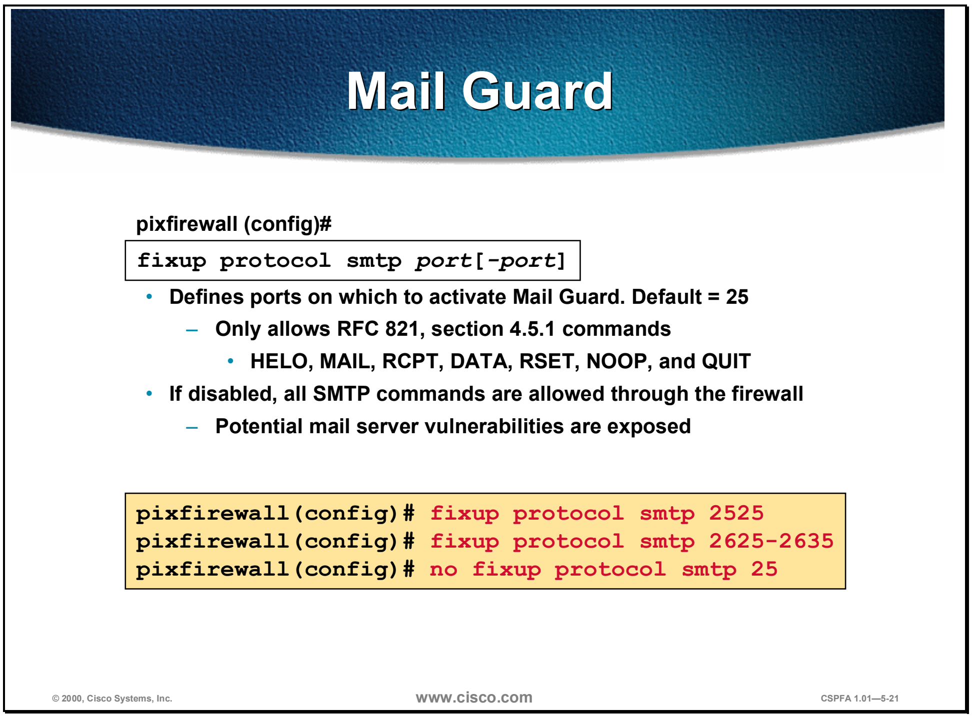
fixup protocol h323 ***port*****[-*****port*****]**

no fixup protocol h323 ***port*****[-*****port*****]**

Argument	Description
***port*[-*port*]**	Single port or port range that the PIX Firewall will inspect for H.323 connections.

Attack Guards

This section discusses the guards put in place against attacks by mail, Domain Name System (DNS), fragmentation, authentication, authorization, and accounting (AAA) and SYN floods.



Mail Guard provides a safe conduit for Simple Mail Transfer Protocol (SMTP) connections from the outside to an inside e-mail server. Mail Guard allows a mail server to be deployed within the internal network without it being exposed to known security problems with some mail server implementations.

Only the SMTP commands specified in RFC 821 section 4.5.1 are allowed to a mail server. This are: HELO, MAIL, RCPT, DATA, RSET, NOOP, and QUIT.

By default, the PIX Firewall inspects port 25 connections for SMTP traffic. If you have SMTP servers using ports other than port 25, you must use the **fixup protocol smtp** command to have the PIX Firewall inspect these other ports for SMTP traffic.

Use the **no** form of the command to disable the inspection of traffic on the indicated port for SMTP connections. If the **fixup protocol smtp** command is not enabled for a given port, then potential mail server vulnerabilities are exposed.

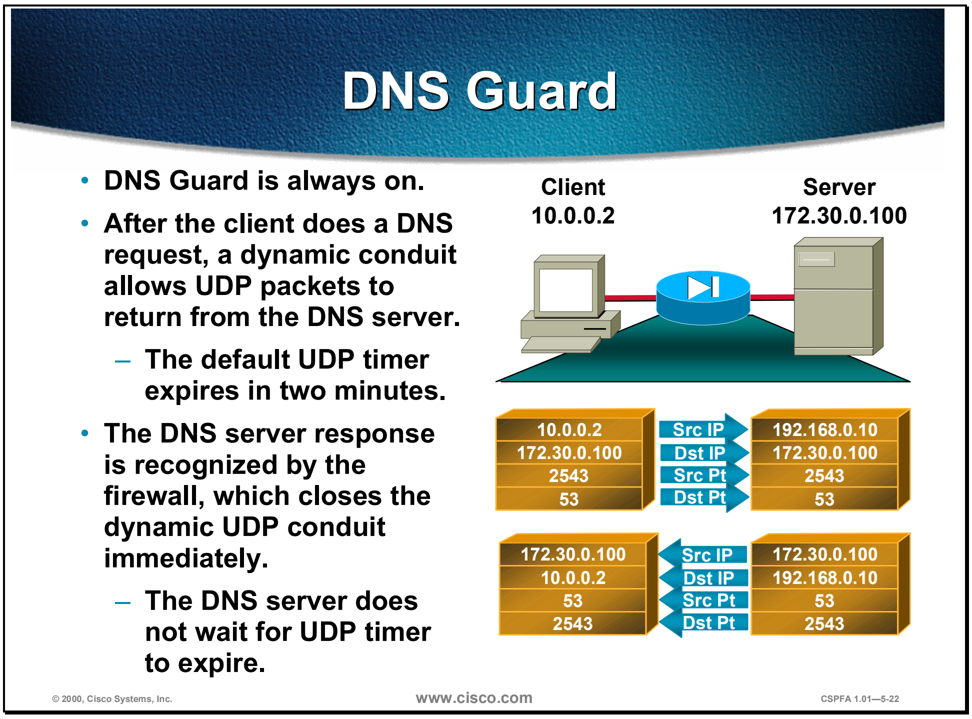
Using the **no fixup protocol smtp** command without any arguments causes the PIX Firewall to clear all previous **fixup protocol smtp** assignments and set port 25 back as the default.

The syntax of the **fixup protocol smtp** command is as follows:

fixup protocol smtp *port*[-*port*]

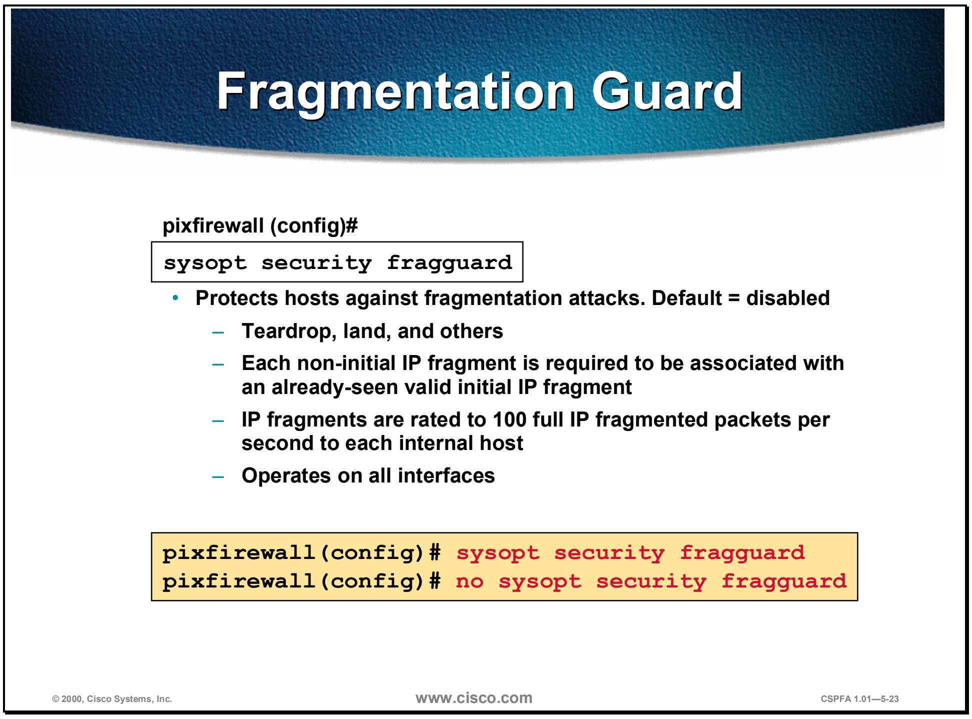
no fixup protocol smtp *port*[-*port*]

Argument	**Description**
***port*[-*port*]**	Single port or port range that the PIX Firewall will inspect for SMTP connections.



DNS Guard identifies an outbound DNS query request and only allows a single DNS response back to the sender. A host may query several servers for a response in case the first server is slow in responding; however, only the first answer to the specific question will be allowed back in. All the additional answers from other servers will be dropped. This feature is always enabled and does the following:

- Automatically tears down the UDP conduit on the PIX Firewall as soon as the DNS response is received. Does not wait for the default UDP timer to close the session.
- Prevents against UDP session hijacking and denial of service (DoS) attacks.



Use the **sysopt security fragguard** command to enable the Fragmentation Guard feature. This feature enforces two addition security checks on IP packets in addition to the security checks recommended by RFC 1858 against the many IP fragment style attacks: teardrop, land, and so on. First, each non-initial IP fragment is required to be associated with an already-seen valid initial IP fragment. Second, IP fragments are rated 100 full IP fragmented packets per second to each internal host. The Fragmentation Guard feature operates on all interfaces in the PIX Firewall and cannot be selectively enabled or disabled by interface.

PIX Firewall uses the **security fragguard command** to enforce the security policy determined by an **access-list permit** or**access -list deny** command to permit or deny packets through the PIX Firewall.

Note Use of the **sysopt security fragguard** command breaks normal IP fragmentation conventions. However, not using this command exposes PIX Firewall protected hosts to the possibility of IP fragmentation attacks. Cisco recommends that packet fragmentation not be permitted on the network if at all possible.

Note If the PIX Firewall is used as a tunnel for FDDI packets between routers, the Fragmentation Guard feature should be disabled.

Note Because Linux sends IP fragments in reverse order, fragmented Linux packets will not pass through the PIX Firewall with the **sysopt security fragguard** enabled.

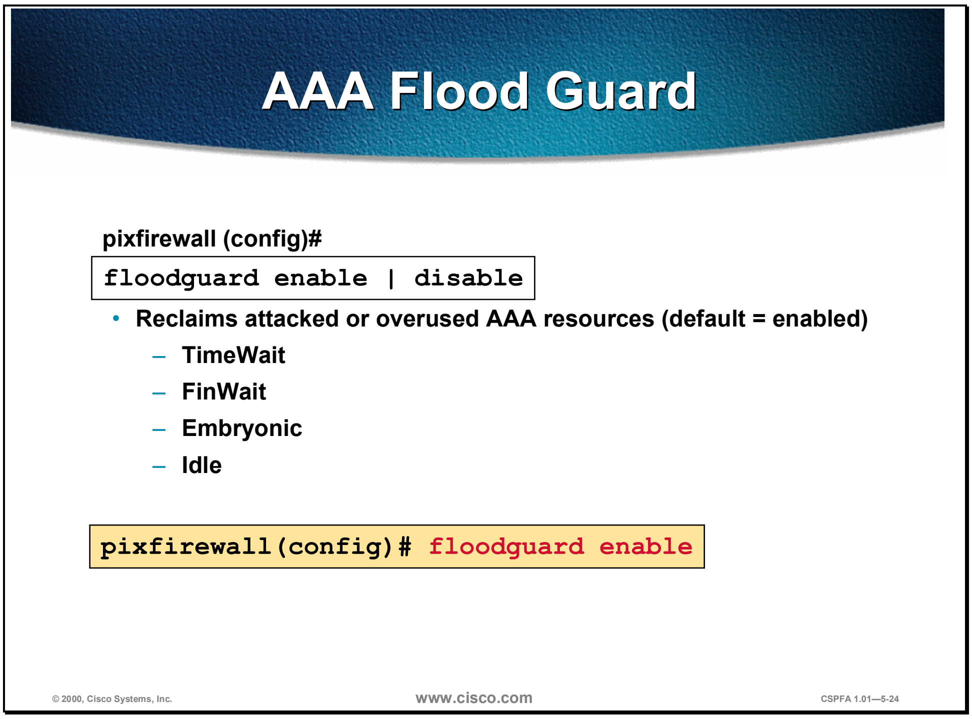
The **sysopt security fragguard** command is disabled by default.

The syntax of the **sysopt security fragguard** command is as follows:

sysopt security fragguard

no sysopt security fragguard

This command has no arguments.



The **floodguard** command lets you reclaim PIX Firewall resources if the user authentication (uauth) subsystem runs out of resources. If an inbound or outbound uauth connection is being attacked or overused, the PIX Firewall actively reclaims TCP user resources. When the resources are depleted, the PIX Firewall lists messages about it being out of resources or out of tcpusers. If the PIX Firewall uauth subsystem is depleted, TCP user resources in different states are reclaimed depending on urgency in the following order:

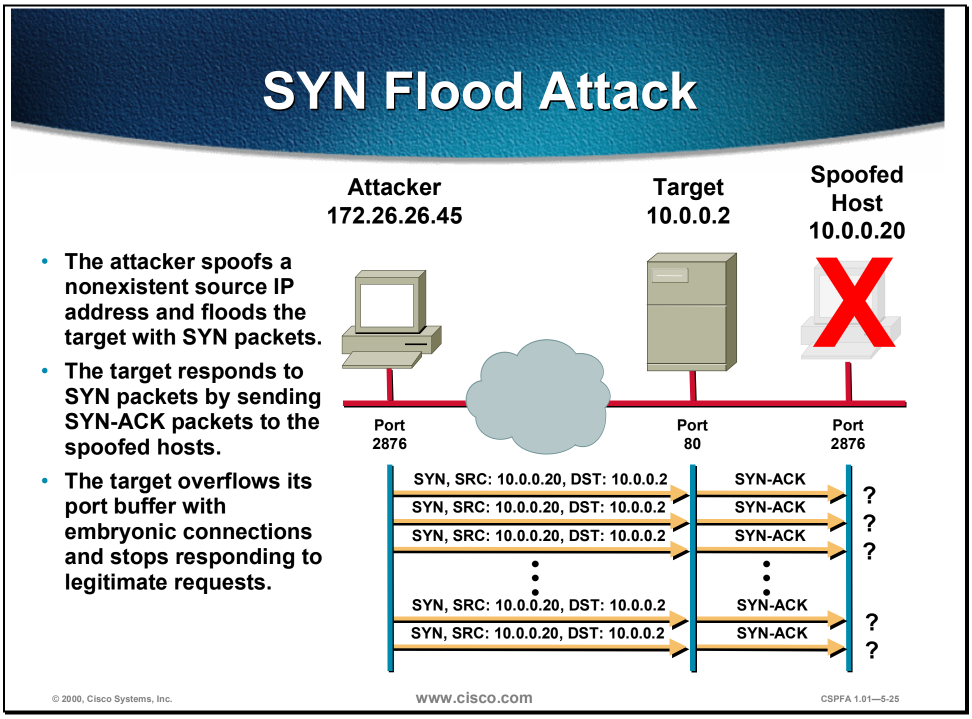
1. Timewait
2. FinWait
3. Embryonic
4. Idle

The **floodguard** command is enabled by default.

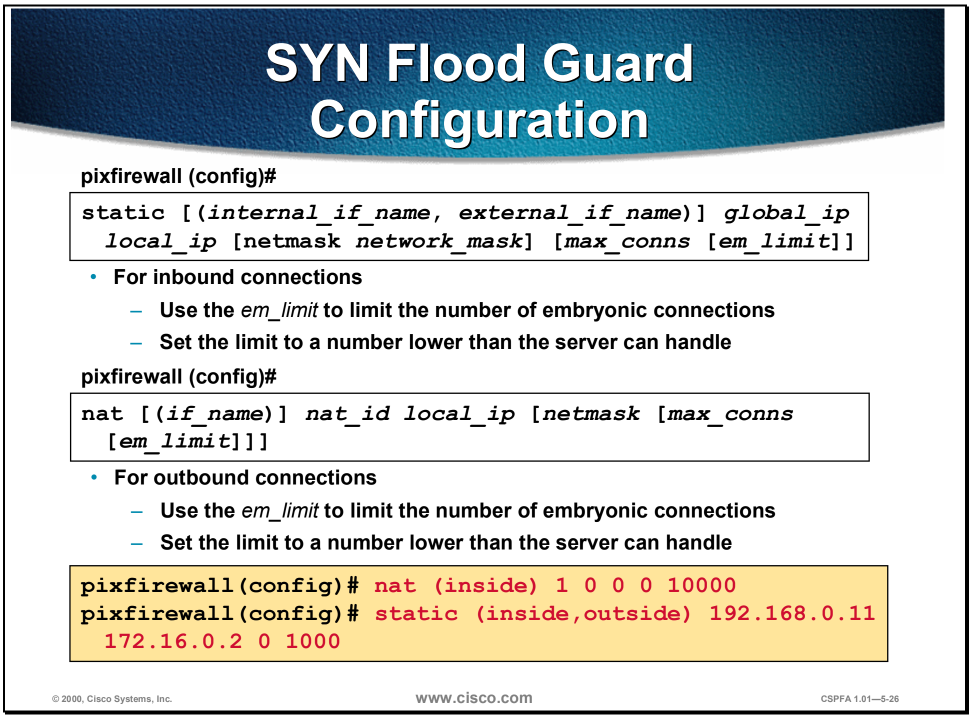
The syntax of the **floodguard** command is as follows:

floodguard enable | disable

Argument	Description
enable	Enable AAA Flood Guard.
disable	Disable AAA Flood Guard.



SYN flood attacks, also known as TCP flood or half-open connections attacks, are common DoS attacks perpetrated against IP servers. The attacker spoofs a nonexistent source IP address or IP addresses on the network of the target host, and floods the target with SYN packets pretending to come from the spoofed host. SYN packets to a host are the first step in the three-way handshake of a TCP-type connection; therefore, the target responds as expected with SYN-ACK packets destined to the spoofed host or hosts. Because these SYN-ACK packets are sent to hosts that do not exist, the target sits and waits for the corresponding ACK packets that never show up. This causes the target to overflow its port buffer with embryonic or half-open connections and stop responding to legitimate requests.



To protect internal hosts against DoS attacks, use the **static** command to limit the number of embryonic connections allowed to the server. Use the ***em_limit*** argument to limit the number of embryonic or half-open connections that the server or servers you are trying to protect can handle without being attacked by a DoS.

The syntax used in the **static** command for enabling the SYN Flood Guard is as follows:

static ***(internal_if_name, external_if_name) global_ip local_ip*** **[netmask** ***network_mask*****]** ***max_conns em_limit***

Argument	Description
internal_if_name	The internal network interface name.
external_if_name	The external network interface name.
global_ip	The global IP address for an outside interface. This address cannot be a PAT IP address.
local_ip	The local IP address on an inside network.
netmask ***network_mask***	The network mask for the ***global_ip*** and ***local_ip***.
max_conns	The maximum connections permitted to the ***local_ip***. The default = 0 (unlimited).
em_limit	The maximum embryonic connection permitted to the ***local_ip***. The default = 0 (unlimited).

To protect external hosts against DoS attacks, and to limit the number of embryonic connections allowed to the server, use the **nat** command. Use the ***em_limit*** argument to limit the number of embryonic or half-open connections that the server or servers you are trying to protect can handle without being attacked by a DoS.

The syntax used in the **nat** command for enabling the SYN Flood Guard is as follows:

nat ***(if_name) nat_id local_ip netmask max_conns em_limit***

Argument	Description
if_name	The internal network interface name.
nat_id	A number used for matching with a corresponding global pool of IP addressees. The matching global pool must used the same ***nat_id***.
local_ip	The internal IP address or networks that will be translated to a global pool of IP addresses.
netmask	The network mask for the ***local_ip***.
max_conns	The maximum connections permitted to hosts accessed from ***local_ip***. The default = 0 (unlimited).
em_limit	The maximum embryonic connection permitted to hosts accessed from ***local_ip***. The default = 0 (unlimited).

Lab: Configure and Test Advanced Protocol Handling and Attack Guards on the Cisco Secure PIX Firewall

Complete the following lab exercise to practice what you have learned in this chapter.

Directions

Your task for this lab exercise is to

- Display the fixup protocol configurations.
- Change the fixup protocol configurations.
- Test the outbound FTP fixup protocol.
- Test the inbound FTP fixup protocol.

Task 1: Display the Fixup Protocol Configurations

Perform the following step and enter the command as directed to see the current configurations of your PIX Firewall:

Step 1 List the fixup protocols that are running on your PIX Firewall:

```
pixfirewall(config)# show fixup protocol
```

Q 1) In the spaces provided, write the ports assigned to all the fixup protocols:

ftp ________________ rsh ________________

http _______________ sqlnet ______________

smtp ______________ rtsp ________________

h323 ______________

Task 2: Change the Fixup Protocol Configurations

Perform the following steps and enter the commands as directed to change some of the current configurations of your PIX Firewall:

Step 1 Disable all fixup protocols except for FTP:

```
pixfirewall(config)# no fixup protocol http 80
pixfirewall(config)# no fixup protocol smtp 25
pixfirewall(config)# no fixup protocol h323 1720
pixfirewall(config)# no fixup protocol rsh 514
pixfirewall(config)# no fixup protocol sqlnet 1521
```

Step 2 Define a port for RTSP connections:


```
pixfirewall(config)# fixup protocol rtsp 554
```

Step 3 Define a range of ports for SQL*Net connections:

```
pixfirewall(config)# fixup protocol sqlnet 66-76
```

Step 4 Verify the fixup protocol settings:

```
pixfirewall(config)# show fixup protocol
```

Step 5 After verifying the fixup protocol settings using the **show fixup protocol** command, fill in the blanks below using the output from this command:

```
pixfirewall(config)# show fixup protocol
fixup protocol http  __________
fixup protocol smtp  __________
fixup protocol h323  __________
fixup protocol rsh   __________
fixup protocol rtsp  __________
fixup protocol sqlnet__________
```

Task 3: Test the Outbound FTP Fixup Protocol

Perform the following steps and enter the commands as directed to test the outbound FTP fixup protocol:

Step 1 Enable console logging on your PIX Firewall:

```
pixfirewall(config)# logging console debug
```

Step 2 Ftp to the backbone server from your workstation using the Windows FTP client:

```
C:\ ftp 172.30.1.50
User (172.30.1.50:(none)): anonymous
Password: user@
```

Step 3 Do a directory listing at the FTP prompt:

```
ftp> dir
```

Q 1) What logging messages were generated on your PIX Firewall console?

____

____

____

____

Step 4 Quit your FTP session:

```
ftp> quit
```

Step 5 Turn off fixup protocol ftp on your PIX Firewall:

```
pixfirewall(config)# no fixup protocol ftp
```

Step 6 Again, ftp to the backbone server from your workstation using the Windows FTP client:

```
C:\ ftp 172.30.1.50
User (172.30.1.50:(none)): anonymous
Password: user@
```

Q 2) Were you able to log into the server? Why or why not?

____

____

____

Step 7 Do a directory listing at the FTP prompt:

```
ftp> dir
```

Q 3) Were you able to see a file listing? Why or why not?

____

____

____

____

Step 8 Quit your FTP session:

```
ftp> quit
```

Note If the FTP client is hung up, press **Ctrl+C** until you break back to the C:\ prompt.

Step 9 Ftp to the backbone server from your workstation using your Web browser. To do this, in the URL field enter

ftp://172.30.1.50

Q 4) Were you able to connect? Why or why not?

____

____

____

Q 5) Were you able to see a file listing? Why or why not?

____

____

____

Step 10 Close your Web browser.

Task 4: Test the Inbound FTP Fixup Protocol

Perform the following steps and enter the commands as directed to test the inbound FTP fixup protocol:

Step 1 Re-enable FTP fixup protocol on your PIX Firewall:

`pixfirewall(config)#` **`fixup protocol ftp 21`**

Step 2 Ftp to a peer pod's inside FTP server from your workstation using your Web browser. To do this, in the URL field enter

`ftp://192.168.Q.11`

Note where Q = peer pod

Note The peer pod number is assigned by the instructor.

Q 1) What logging messages were generated on your PIX Firewall console?

____

____

____

____

Step 3 Close your Web browser.

Step 4 Turn off the FTP fixup protocol on your PIX Firewall:

`pixfirewall(config)#` **`no fixup protocol ftp`**

Step 5 Ftp to a peer pod's inside FTP server from your workstation using your Web browser. To do this, in the URL field enter

`ftp://192.168.Q.10` (where Q=peer pod)

Note	The peer pod number is assigned by the instructor.

Q 2) Were you able to connect to the peer pod's inside FTP server? Why or why not?

____

____

____

Q 3) Were you able to see the file listings? Why or why not?

____

____

____

Task 5: Set the Fixup Protocols to Default

Perform the following steps and enter the commands as directed to set all fixups to the factory default:

Step 1 Set all fixup protocols to the factory defaults:

```
pixfirewall(config)# clear fixup
```

Step 2 Verify the fixup protocol settings:

```
pixfirewall(config)# show fixup protocol
fixup protocol ftp 21
fixup protocol http 80
fixup protocol smtp 25
fixup protocol h323 1720
fixup protocol rsh 514
fixup protocol sqlnet 1521
```


Summary

This section summarizes what you learned in this chapter.

Summary

- The PIX Firewall uses special handling for the following advanced protocols: FTP, rsh, and SQL*Net.
- FTP, rsh, and SQL*Net Fixup protocols can be configured for ports other than the defaults.
- The PIX Firewall handles the following multimedia protocols: RTSP and H.323.
- RTSP and H.323 Fixup protocols can be configured for ports other that the defaults.
- The PIX Firewall has the following attack guards to help protect systems from malicious attacks: Mail Guard, DNS Guard, Fragmentation Guard, AAA Flood Guard, and SYN Flood Defender.

 www.cisco.com CSPFA 1.01—5-28

Cisco Secure PIX Firewall Failover

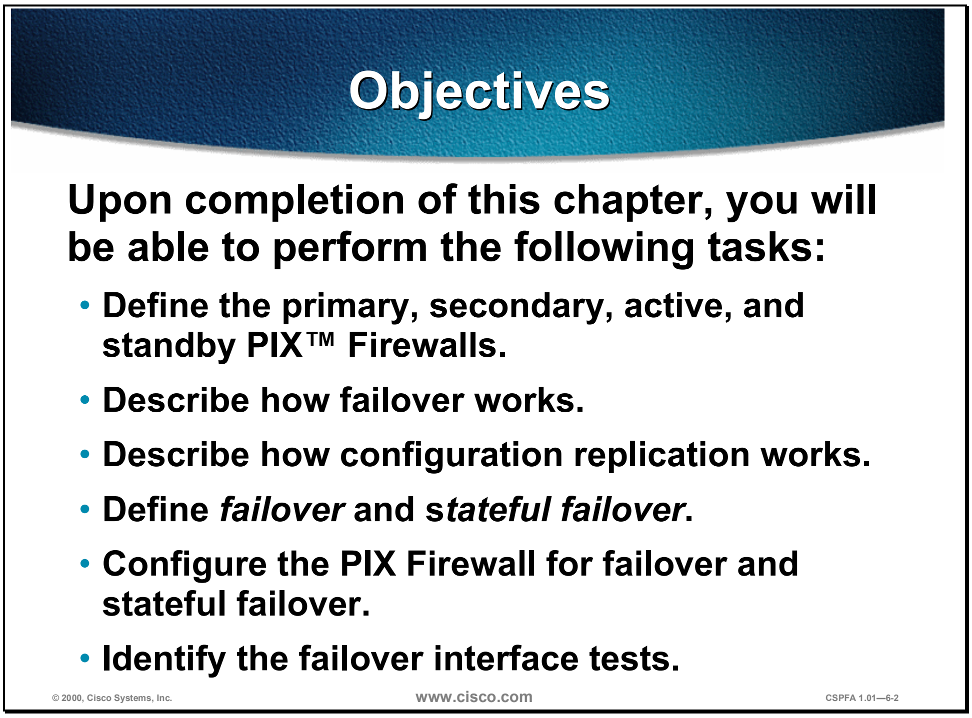
Overview

This chapter includes the following topics:

- Objectives
- Understand failover
- Configure failover
- Lab exercise
- Summary

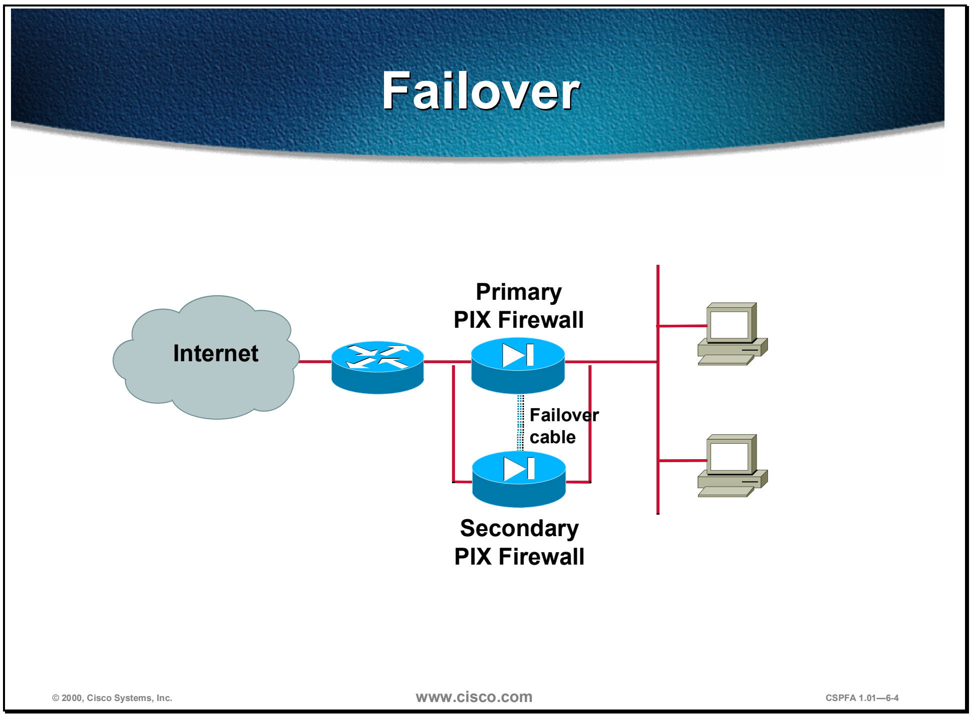
Objectives

This section lists the chapter's objectives.



Understand Failover

This section discusses what failover is and how it works.



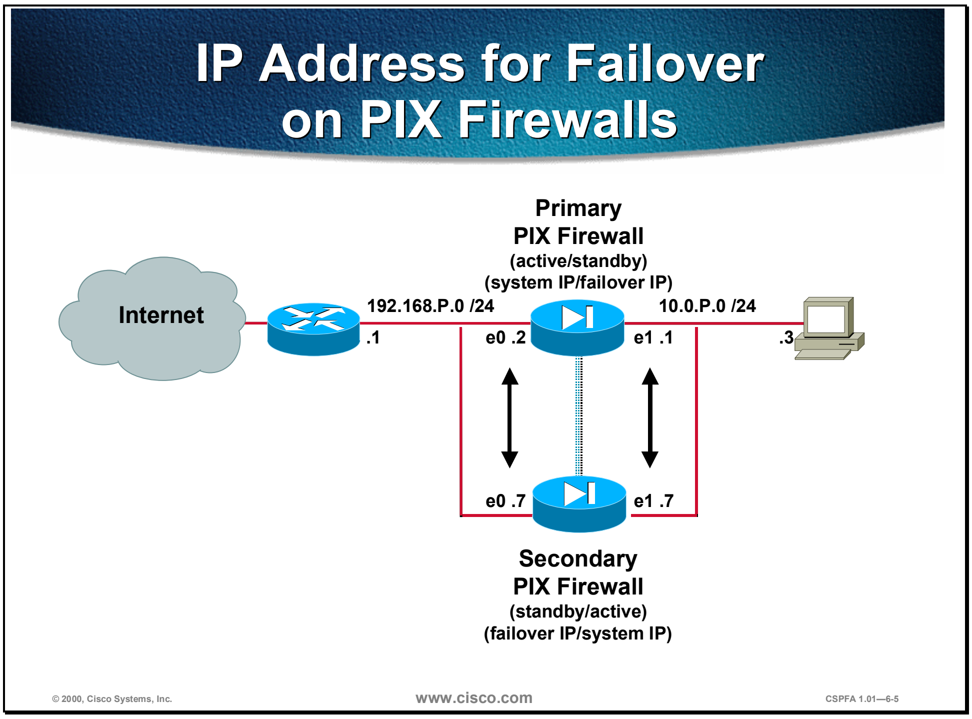
The failover function for the Cisco Secure PIX Firewall™ provides a safeguard in case a PIX Firewall fails. Specifically, when one PIX Firewall fails, another immediately takes its place.

In the failover process, there are two PIX Firewalls: the primary PIX Firewall and the secondary PIX Firewall. The primary PIX Firewall functions as the active PIX Firewall, performing normal network functions. The secondary PIX Firewall functions as the standby PIX Firewall, ready to take control should the active PIX Firewall fail to perform. When the primary PIX Firewall fails, the secondary PIX Firewall becomes active while the primary PIX Firewall goes on standby. This entire process is called *failover*.

The primary PIX Firewall is connected to the secondary PIX Firewall through a failover connection: a failover cable. The failover cable has one end labeled *primary*, which plugs into the primary PIX Firewall, and the other end labeled *secondary*, which plugs into the secondary PIX Firewall.

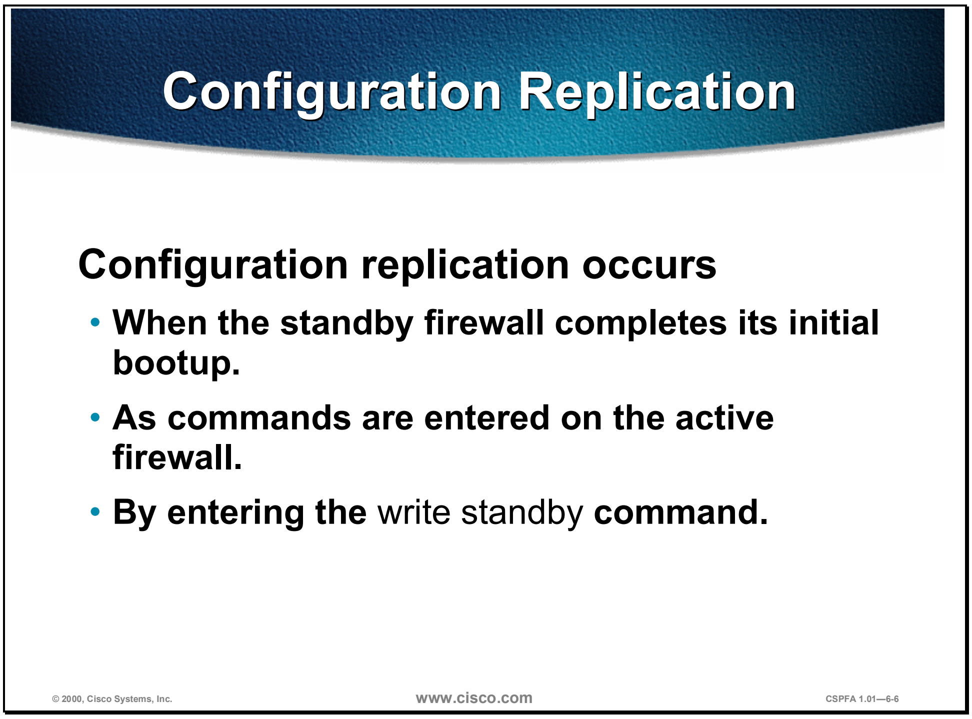
A failover occurs when one of the following situations takes place:

- A power-off or a power-down condition occurs on the active PIX Firewall
- The active PIX Firewall is rebooted
- A link goes down on the active PIX Firewall for more than 30 seconds
- The message, "Failover active" occurs on the standby PIX Firewall
- Block memory exhaustion occurs for 15 consecutive seconds or more on the active PIX Firewall



When actively functioning, the primary PIX Firewall uses system IP addresses and MAC addresses. The secondary PIX Firewall, when on standby, uses failover IP addresses and MAC addresses.

When the primary PIX Firewall fails and the secondary PIX Firewall becomes active, the secondary PIX Firewall assumes the system IP addresses and MAC addresses of the primary PIX Firewall. Then the primary PIX Firewall, functioning in standby, assumes the failover IP addresses and MAC addresses of the secondary PIX Firewall.



Configuration replication is when the configuration of the primary PIX Firewall is replicated to the secondary PIX Firewall. To perform configuration replication, both the primary and secondary PIX Firewalls must be configured exactly the same and running the same software release. Configuration replication occurs over the failover cable from the active PIX Firewall to the standby PIX Firewall in three ways:

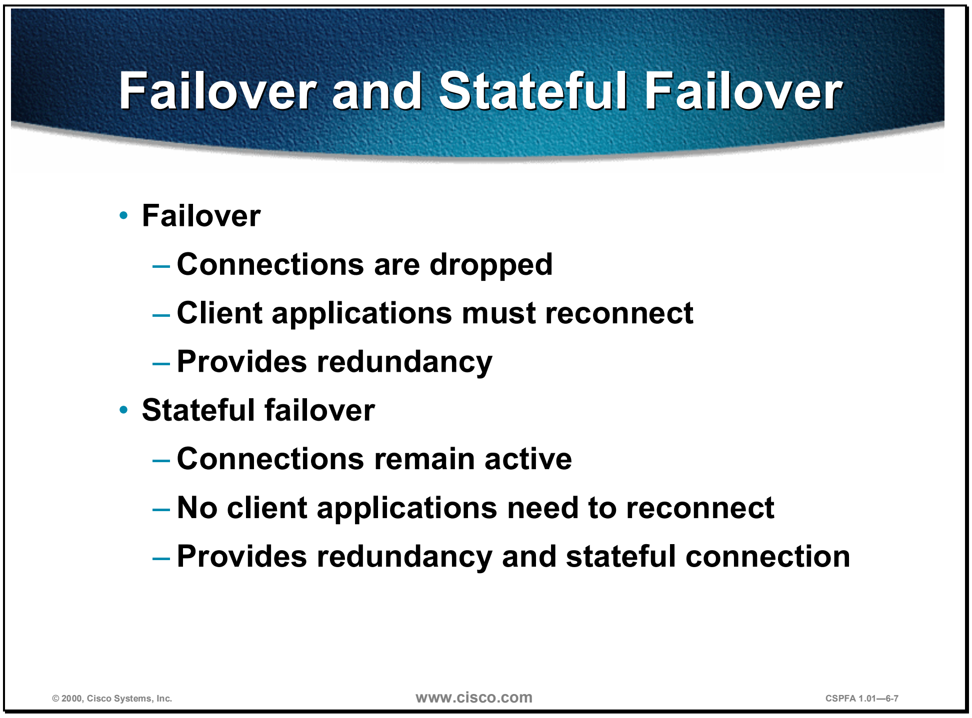
- When the standby PIX Firewall completes its initial bootup, the active PIX Firewall replicates its entire configuration to the standby PIX Firewall.
- As commands are entered on the active PIX Firewall, they are sent across the failover cable to the standby PIX Firewall.
- By entering the **write standby** command on the active PIX Firewall, which forces the entire configuration in memory to be sent to the standby PIX Firewall.

Configuration replication only occurs from memory to memory. Because this is not a permanent place to store configurations, you must use the **write memory** command to write the configuration into Flash memory. If a failover occurs during the replication, the new active PIX Firewall will have only a partial configuration. The newly active PIX Firewall will then reboot itself to recover the configuration from the Flash memory or resynchronize with the newly standby PIX Firewall.

When replication starts, the PIX Firewall console displays the message “Sync Started”, and when complete, displays the message “Sync Completed”. During replication, information cannot be entered on the PIX Firewall console. Replication can take a long time to complete for a large configuration because the failover cable is used.

Configure Failover

This section discusses what failover and stateful failover modes are, and how to configure stateful failover.



As stated earlier in the chapter, failover enables the standby PIX Firewall to take over the duties of the active PIX Firewall when the active PIX Firewall fails. There are two types of failover:

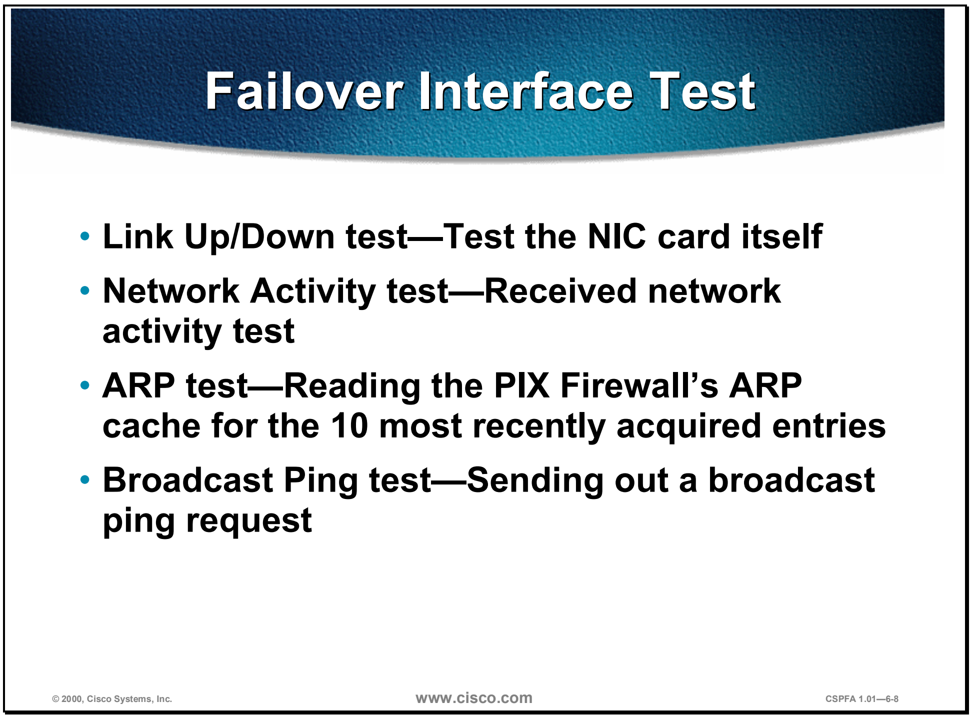
- Failover—When the active PIX Firewall fails and the standby PIX Firewall becomes active, all connections are lost and client applications must perform a new connection to restart communication through the PIX Firewall. The disconnection happens because the active PIX Firewall does not pass the stateful connection information to the standby PIX Firewall.
- Stateful failover—When the active PIX Firewall fails and the standby PIX Firewall becomes active, the same connection information is available at the new active PIX Firewall, and end-user applications are not required to do a reconnect to keep the same communication session. The connections remain because the stateful failover feature passes per-connection stateful information to the standby PIX Firewall.

Stateful failover requires an 100 Mbps Ethernet interface to be used exclusively for passing state information between the two PIX Firewalls. This interface can be connected to any of the following:

- Category 5 crossover cable directly connecting the primary PIX Firewall to the secondary PIX Firewall
- 100BaseTX half-duplex hub using straight Category 5 cables

- 100BaseTX full duplex on a dedicated switch or dedicated virtual LAN (VLAN) of a switch

Note The PIX Firewall does not support the use of either Token Ring or FDDI for the stateful failover dedicated interface. Data is passed over the dedicated interface using IP protocol 105. No hosts or routers should be on this interface.



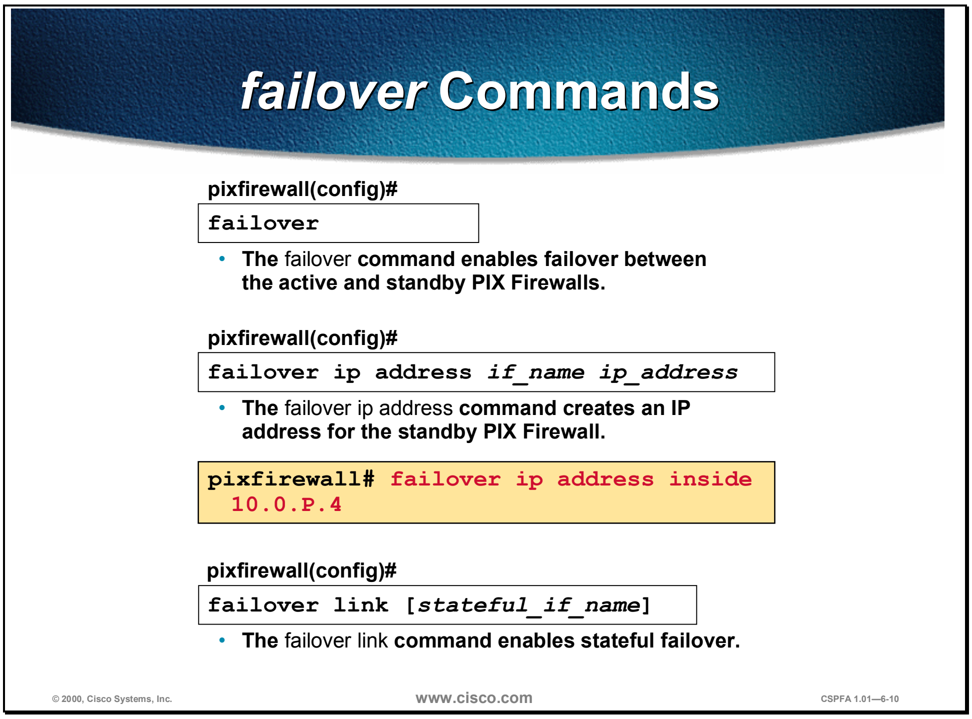
Both the primary and secondary PIX Firewalls send special failover “hello” packets to each other over all network interfaces and the failover cable every 15 seconds to make sure that everything is working. When a failure occurs in the active PIX Firewall, and it is not because of a loss of power in the standby PIX Firewall, failover begins a series of tests to determine which PIX Firewall has failed. The purpose of these tests is to generate network traffic to determine which (if either) PIX Firewall has failed.

At the start of each test, each PIX Firewall clears its received packet count for its interfaces. At the conclusion of each test, each PIX Firewall looks to see if it has received any traffic. If it has, the interface is considered operational. If one PIX Firewall receives traffic for a test and the other PIX Firewall does not, the PIX Firewall that did not receive traffic is considered failed. If neither PIX Firewall has received traffic, then the tests continue.

The following are the four different tests used to test for failover:

- LinkUp/Down—This is a test of the NIC card itself. If an interface card is not plugged into an operational network, it is considered failed (for example, the hub or switch has failed, has a failed port, or a cable is unplugged). If this test does not find anything, the Network Activity test begins.
- Network Activity—This is a received network activity test. The PIX Firewall counts all received packets for up to five seconds. If any packets are received at any time during this interval, the interface is considered operational and testing stops. If no traffic is received, the ARP test begins.
- ARP—The ARP test consists of reading the PIX Firewall’s ARP cache for the ten most recently acquired entries. The PIX Firewall sends ARP requests one at a time to these machines, attempting to stimulate network traffic. After each request, the PIX Firewall counts all received traffic for up to five seconds. If traffic is received, the interface is considered operational. If no traffic is received, an ARP request is sent to the next machine. If at the end of the list no traffic has been received, the ping test begins.

- Broadcast Ping—The ping test consists of sending out a broadcast ping request. The PIX Firewall then counts all received packets for up to five seconds. If any packets are received at any time during this interval, the interface is considered operational and testing stops. If no traffic is received, the testing starts over again with the ARP test.



Use the **failover** command to enable failover between two PIX Firewalls. The syntax for the **failover** command is as follows:

failover

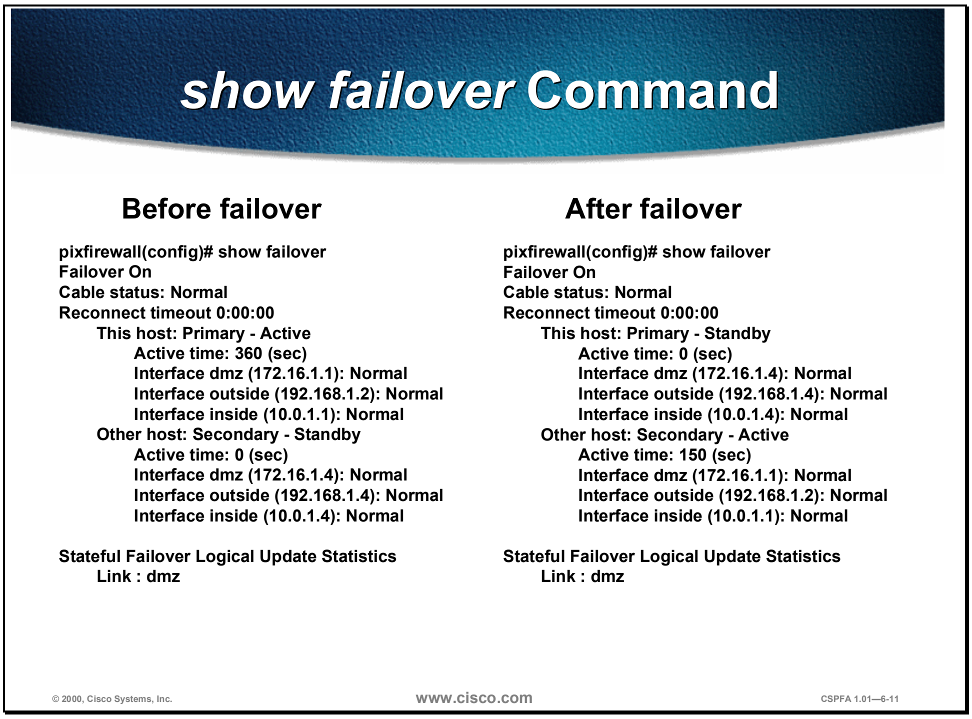
Use the **failover ip address** command to configure the failover IP address for the standby PIX Firewall. The syntax for the **failover ip address** command is as follows:

failover ip address ***if_name ip_address***

The **failover link** command enables stateful failover. The syntax for the **failover link** command is as follows:

failover link [*stateful_if_name*]

Argument	Description
active	Makes a PIX Firewall the active PIX Firewall. Use this command when you need to force control of the connection back to the unit you are accessing, such as when you want to switch control back from a PIX Firewall after you have fixed a problem and want to restore service to the primary PIX Firewall.
if_name	Interface on which the standby PIX Firewall resides.
ip_address	The IP address used by the standby PIX Firewall to communicate with the active PIX Firewall.
link	Specifies the interface where a fast LAN link is available for stateful failover.
stateful_if_name	In addition to the failover cable, a dedicated fast LAN link is required to support stateful failover. The default interface is the highest LAN port with failover configured.



The figure above is an example of the **show failover** command both before and after failure to the primary PIX Firewall. This example shows the primary PIX Firewall going from active mode to standby mode and the secondary PIX Firewall going from standby mode to active mode during a failover. During this process, the primary PIX Firewall swaps its system IP addresses with the secondary PIX Firewall's failover IP addresses.

Lab Exercise:

Complete the following lab exercise to practice what you learned in this chapter.

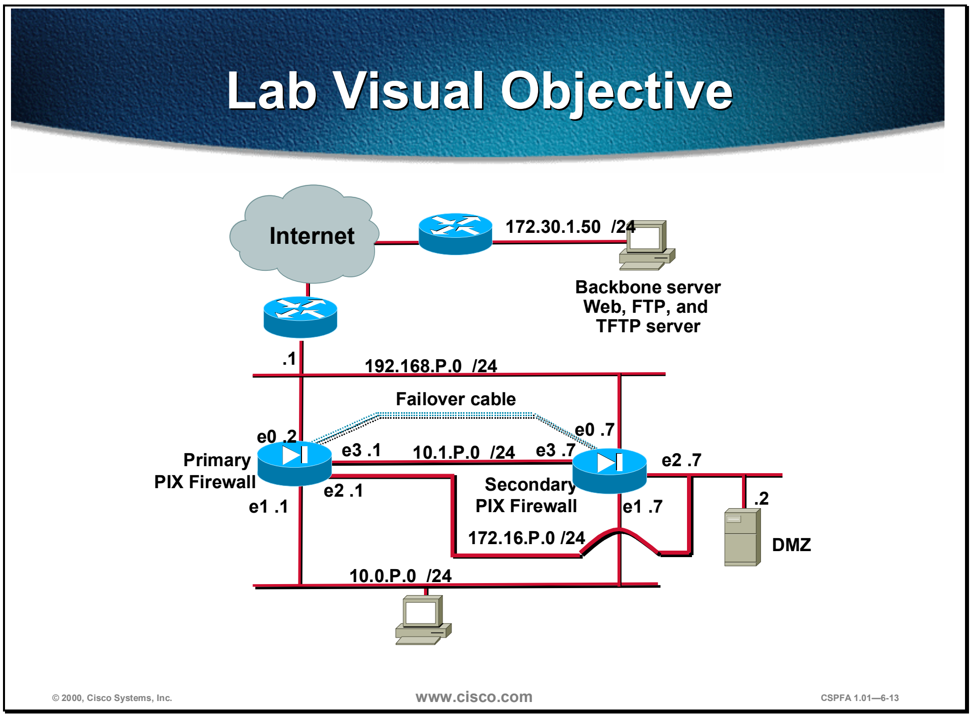
Objectives

In this lab exercise you will complete the following tasks:

- Configure the primary PIX Firewall for failover to the secondary PIX Firewall
- Make the primary PIX Firewall active
- Configure the primary PIX Firewall for stateful failover

Visual Objectives

The following figure displays the configuration you will complete in this lab exercise.



Task 1: Configure the Primary PIX Firewall for Failover to the Secondary PIX Firewall

Perform the following lab steps to configure the primary PIX Firewall for failover to the secondary PIX Firewall:

Step 1 Enter the **configure terminal** command to enter into config mode:

```
pixfirewall> config terminal
```

Step 2 Configure another interface for Stateful failover, for later in Task 3.

Step 3 Assign the PIX Firewall with a Failover interface a name (failover) and security level (55)

```
pixfirewall (config)# nameif e3 MYFAILOVER security55
```

Step 4 Enable the interface for an Intel full duplex.

```
pixfirewall (config)# interface e3 100full
```

Step 5 Assign a IP address to the interface.

```
pixfirewall (config)# ip address MYFAILOVER 10.1.P.1
```

(where P = pod number, and Q = peer pod number)

Step 6 Save all changes to flash memory:

```
pixfirewall (config)# write memory
```

Step 7 Test connections to 172.30.1.50 by using FTP and HTTP connections.

Step 8 Make sure that you are connected to the Primary PIX Firewall. Enter the **failover** command to enable failover:

```
pixfirewall (config)# failover
```

Step 9 Make sure that the Primary PIX Firewall is enabled by using the **show failover** command:

```
pixfirewall (config)# show failover
```

Step 10 Enter the **failover ip address** command to configure the primary PIX Firewall with the secondary PIX Firewall IP addresses for each interface that is being used:

```
pixfirewall (config)# failover ip address outside 192.168.P.7
pixfirewall (config)# failover ip address inside 10.0.P.7
pixfirewall (config)# failover ip address dmz 172.16.P.7
pixfirewall (config)# failover ip address MYFAILOVER 10.1.P.7
```

Step 11 Write the configuration to Flash memory:

```
pixfirewall(config) write memory
```

Step 12 Connect the failover cable to the Primary PIX Firewall making sure to use the Primary end of the cable.

Step 13 Connect the other end of the failover cable marked *Secondary* to the Secondary PIX Firewall.

Step 14 Power up the Secondary PIX Firewall so that the replication of information from the Primary PIX Firewall to the Secondary PIX Firewall can occur.

Step 15 After the Secondary PIX Firewall is operational, enter the **show failover** command on the Primary PIX Firewall to make sure that the replication is complete and that communication between the PIX Firewalls is working:

```
pixfirewall (config)# show failover
```

```
pixfirewall(config)# show failover
Failover On
Cable status: Normal
Reconnect timeout 0:00:00
        This host: Primary - Active
                Active time: 7350 (sec)
                Interface pix/intf5 (127.0.0.1): Link Down (Waiting)
                Interface pix/intf4 (127.0.0.1): Link Down (Waiting)
                Interface MYFAILOVER (10.1.P.1): Normal
                Interface dmz (172.16.P.1): Normal
                Interface outside (192.168.P.2): Normal
                Interface inside (10.0.P.1): Normal
        Other host: Secondary - Standby
                Active time: 0 (sec)
                Interface pix/intf5 (0.0.0.0): Link Down (Waiting)
                Interface pix/intf4 (0.0.0.0): Link Down (Waiting)
                Interface MYFAILOVER (10.1.P.7): Normal
                Interface dmz (172.16.P.7): Normal
                Interface outside (192.168.P.7): Normal
                Interface inside (10.0.P.7): Normal
```

Step 16 Test connections to 172.30.1.50 by using FTP.

Step 17 Log back in to your FTP server.

Step 18 Power down the primary PIX Firewall to test failover. This ensures that the active PIX Firewall has switched from the primary PIX Firewall to the secondary PIX Firewall.

Step 19 To verify that the Secondary PIX Firewall is active, enter the **show failover** command:

```
pixfirewall (config)# show failover
Failover On
Cable status: Normal
Reconnect timeout 0:00:00
        This host: Primary - Standby
                Active time: 0 (sec)
                Interface pix/intf5 (127.0.0.1): Link Down (Waiting)
                Interface pix/intf4 (127.0.0.1): Link Down (Waiting)
                Interface MYFAILOVER (10.1.P.7): Normal
                Interface dmz (172.16.P.7): Normal
                Interface outside (192.168.P.7): Normal
                Interface inside (10.0.P.7): Normal
        Other host: Secondary - Active
                Active time: 7350 (sec)
                Interface pix/intf5 (0.0.0.0): Link Down (Waiting)
```

```
            Interface pix/intf4 (0.0.0.0): Link Down (Waiting)
            Interface MYFAILOVER (10.1.P.1): Normal
            Interface dmz (172.16.P.1): Normal
            Interface outside (192.168.P.2): Normal
            Interface inside (10.0.P.1): Normal
```

Step 20 Test connections to 172.30.1.50 by using FTP.

Step 21 Log back in to your FTP server.

Task 2: Make the Primary PIX Firewall Active

Perform the following lab steps to make the primary PIX Firewall the active PIX Firewall:

Step 1 Make the primary PIX Firewall the active PIX Firewall by using the **failover active** command. Make sure that you are connected to the primary PIX Firewall's console port.

```
pixfirewall (config)# failover active
```

Step 2 Verify that the failover active command worked by using the show failover command. The Primary PIX Firewall should show that it is in active mode and the Secondary PIX Firewall should show that it is in the standby mode.

```
pixfirewall (config)# show failover
Failover On
Cable status: Normal
Reconnect timeout 0:00:00
        This host: Primary - Active
                Active time: 525 (sec)
                Interface pix/intf5 (127.0.0.1): Link Down (Waiting)
                Interface pix/intf4 (127.0.0.1): Link Down (Waiting)
                Interface MYFAILOVER (10.1.P.1): Normal
                Interface dmz (172.16.P.1): Normal
                Interface outside (192.168.P.2): Normal
                Interface inside (10.0.P.1): Normal
        Other host: Secondary - Standby
                Active time: 2300 (sec)
                Interface pix/intf5 (0.0.0.0): Link Down (Waiting)
                Interface pix/intf4 (0.0.0.0): Link Down (Waiting)
                Interface MYFAILOVER (10.1.P.7): Normal
                Interface dmz (172.16.P.7): Normal
                Interface outside (192.168.P.7): Normal
                Interface inside (10.0.P.7): Normal
```

Task 3: Configure the Primary PIX Firewall for Stateful Failover

Perform the following lab steps to configure the primary PIX Firewall for stateful failover:

Step 1 Configure the primary PIX Firewall for stateful failover to the secondary PIX Firewall by using the **failover link** command:

```
pixfirewall (config)# failover link MYFAILOVER
```

Step 2 Make sure that the secondary PIX Firewall has the latest changes to the configuration by using the **write memory** command. This will sync up the configuration on both firewalls:

```
pixfirewall (config)# write memory
```

Step 3 Verify that stateful failover is in place by using the **show failover** command:

```
pixfirewall (config)# show failover

Failover On
Cable status: Normal
Reconnect timeout 0:00:00
        This host: Primary - Active
                Active time: 525 (sec)
                Interface pix/intf5 (127.0.0.1): Link Down (Waiting)
                Interface pix/intf4 (127.0.0.1): Link Down (Waiting)
                Interface MYFAILOVER (10.1.1.1): Normal
                Interface dmz (172.16.1.1): Normal
                Interface outside (192.168.1.2): Normal
                Interface inside (10.0.1.1): Normal
        Other host: Secondary - Standby
                Active time: 0 (sec)
                Interface pix/intf5 (0.0.0.0): Link Down (Waiting)
                Interface pix/intf4 (0.0.0.0): Link Down (Waiting)
                Interface MYFAILOVER (10.1.1.7): Normal
                Interface dmz (172.16.1.7): Normal
                Interface outside (192.168.1.7): Normal
                Interface inside (10.0.1.7): Normal

Stateful Failover Logical Update Statistics
        Link : failover
        Stateful Obj    xmit       xerr       rcv        rerr
        General         84         0          82         0
        sys cmd         84         0          80         0
        up time         0          0          2          0
        xlate           0          0          0          0
        tcp conn        0          0          0          0
        udp conn        0          0          0          0
        ARP tbl         0          0          0          0
        RIP Tbl         0          0          0          0

        Logical Update Queue Information
                        Cur     Max     Total
        Recv Q:         0       1       84
        Xmit Q:         0       1       86
```

Step 4 Test the stateful failover to 172.30.1.50 by using the FTP command from the NT server. Login in to the FTP server username **anonymous**, password **cisco**.

Step 5 To download a zip file from the FTP server you just logged into, enter the following command at the FTP prompt:

```
ftp> mget getme.zip
```

Step 6 Start a continuous ping to 172.30.1.50 by using the following command in the command prompt:

```
C:\ ping 172.30.1.50 –t
```

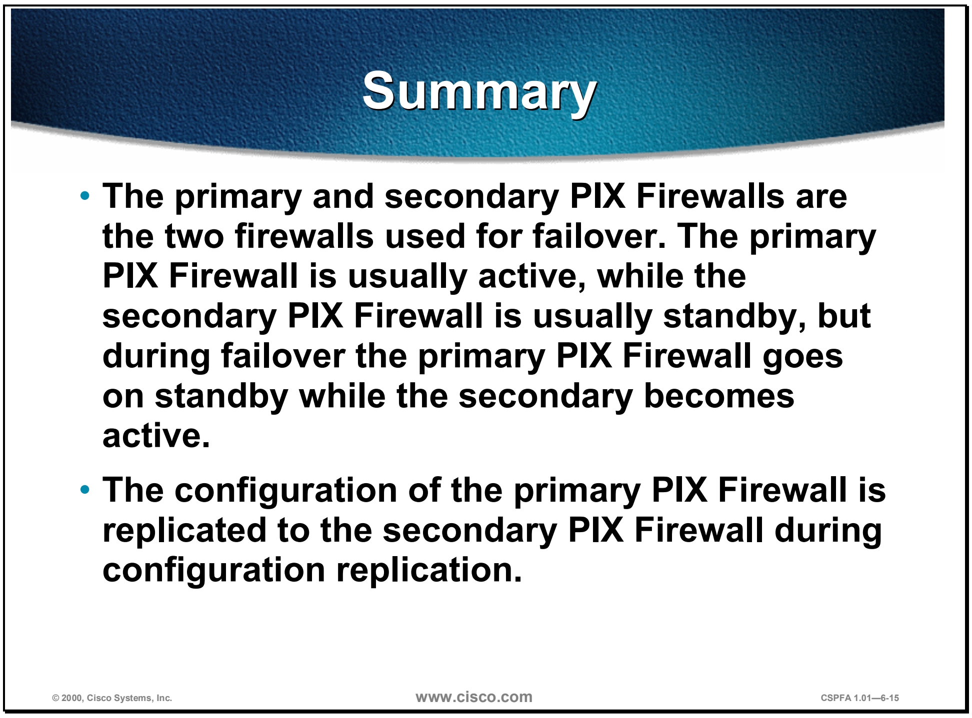
Step 7 Reload the primary PIX Firewall:

```
pixfirewall(config)# reload
```

Step 8 When asked to confirm the reload, press **Enter**.

Summary

This section summarizes what you learned in this chapter.



Summary (cont.)

- During failover, connections are dropped, while during stateful failover, connections remain active.
- There are four interface tests to ensure that the PIX Firewall’s are running:
  - Link Up/Down test
  - Network Activity test
  - ARP test
  - Broadcast Ping test

 www.cisco.com CSPFA 1.01—6-16

7

VPN Configuration with the Cisco Secure PIX Firewall

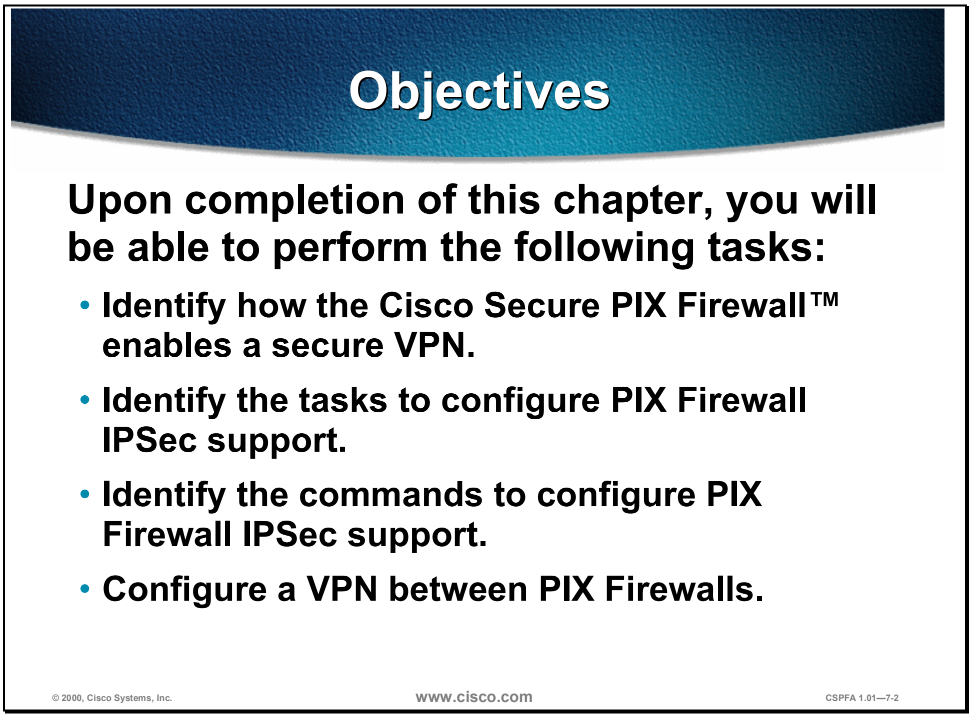
Overview

This chapter includes the following topics:

- Objectives
- The PIX Firewall enables a secure VPN
- IPSec configuration tasks
- Scale PIX Firewall VPNs
- PIX Firewall with CA enrollment
- Lab exercise
- Summary

Objectives

This section lists the chapter's objectives.

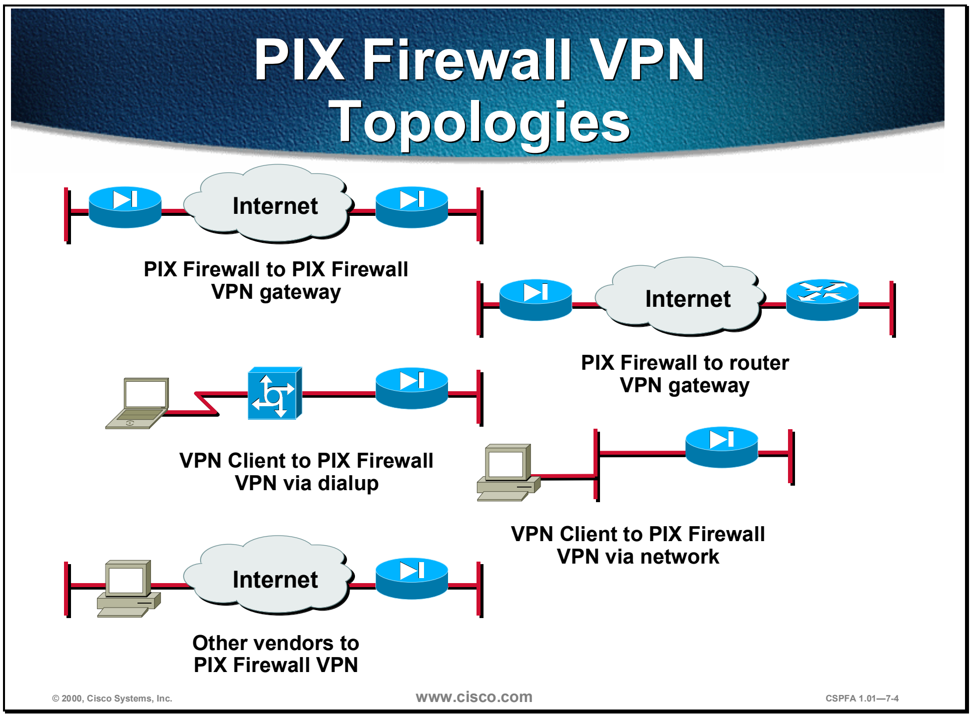


The Cisco Secure PIX Firewall™ Enables a Secure VPN

A Virtual Private Network (VPN) is a service offering secure, reliable connectivity over a shared, public network infrastructure such as the Internet. Because the infrastructure is shared, connectivity can be provided at lower cost than existing dedicated private networks.

The PIX Firewall is a powerful enabler of VPN services. The PIX Firewall's high performance, conformance to open standards, and ease of configuration make it a versatile VPN gateway.

The PIX Firewall VPN chapter covers the basics of IPSec and PIX Firewall VPNs with a focus on PIX Firewall gateway to PIX Firewall gateway communications.



The PIX Firewall enables VPNs in several topologies, as illustrated in the figure:

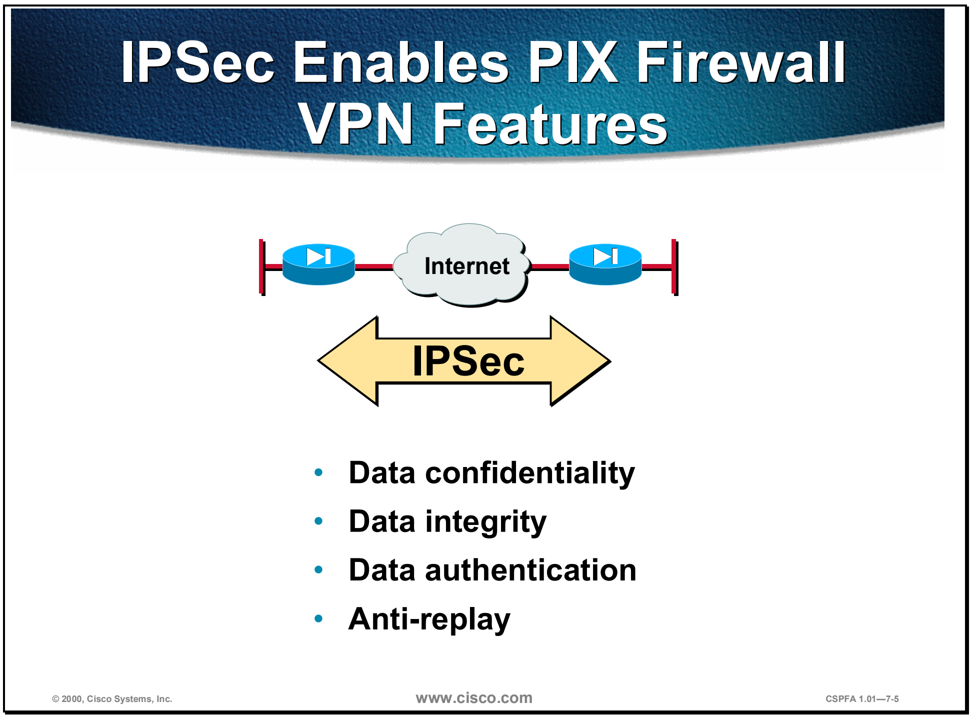
- PIX to PIX secure VPN gateway—Two or more PIX Firewalls can enable a VPN, which secures traffic from devices behind the PIX Firewalls. The secure VPN gateway topology prevents the user from having to implement VPN devices or software inside the network, making the secure gateway transparent to users.
- PIX to Cisco IOS™ router secure VPN gateway—The PIX Firewall and Cisco router, running Cisco Secure VPN software, can interoperate to create a secure VPN gateway between networks.
- Cisco Secure VPN Client to PIX via dialup—The PIX Firewall can become a VPN endpoint for the Cisco Secure VPN Client over a dialup network. The dialup network can consist of ISDN, Public Switched Telephone Network

(PSTN) (analog modem), or digital subscriber line (DSL) communication channels.

- Cisco Secure VPN Client to PIX via network—The PIX Firewall can become a VPN endpoint for the Cisco Secure VPN Client over an IP network.
- Other vendor products to PIX—Products from other vendors can connect to the PIX Firewall if they conform to open VPN standards.

A VPN itself can be constructed in a number of scenarios. The most common are as follows:

- Internet VPN—A private communications channel over the public access Internet. This type of VPN can be divided into the following:
  - Connecting remote offices across the Internet.
  - Connecting remote dial users to their home gateway via an Internet Service Provider (ISP) (sometimes called a Virtual Private Dial Network or VPDN).
- Intranet VPN—A private communication channel within an enterprise or organization that may or may not involve traffic traversing a WAN.
- Extranet VPN—A private communication channel between two or more separate entities that may involve data traversing the Internet or some other WAN.
- In all cases the VPN or tunnel consists of two endpoints that may be represented by PIX Firewalls, Cisco routers, individual client workstations running the Cisco Secure VPN Client, or other vendors' VPN products that conform to open standards.

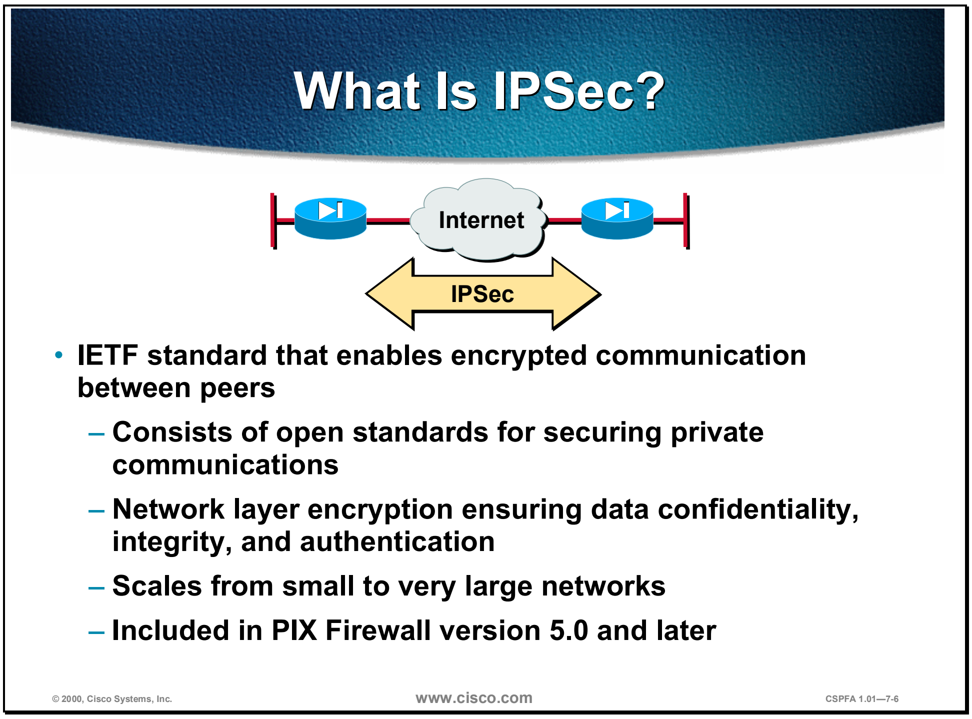


The PIX 5.1 Firewall uses the industry-standard IP Security (IPSec) protocol suite to enable advanced VPN features. The PIX IPSec implementation is based on Cisco IOS IPSec that runs in Cisco routers.

IPSec provides a mechanism for secure data transmission over IP networks, ensuring confidentiality, integrity, and authenticity of data communications over unprotected networks such as the Internet.

IPSec enables the following PIX Firewall VPN features:

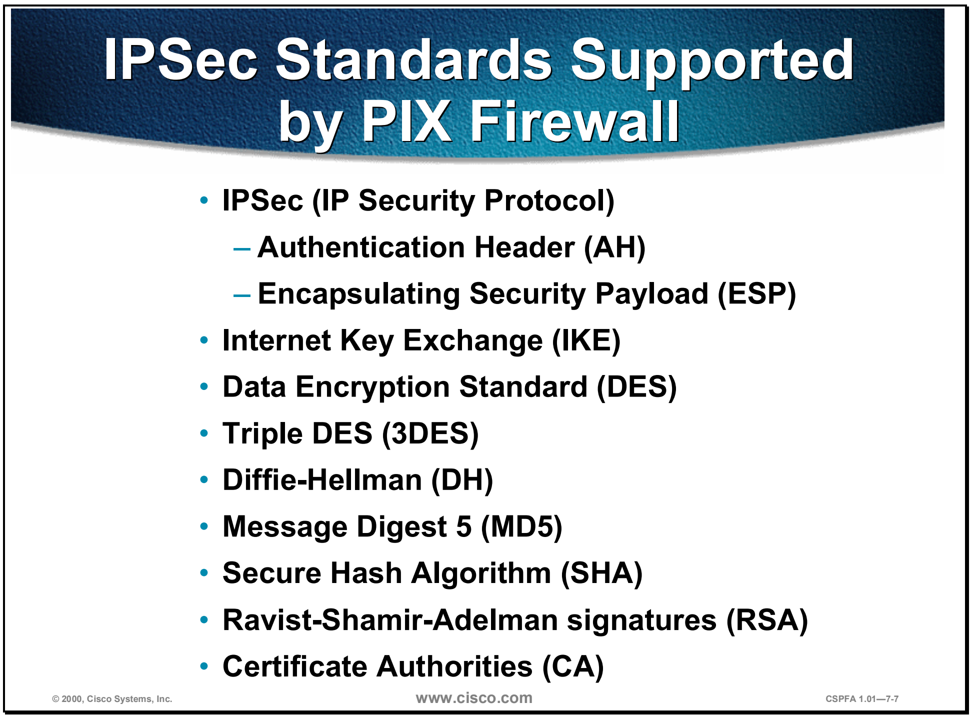
- Data confidentiality—The IPSec sender can encrypt packets before transmitting them across a network.
- Data integrity—The IPSec receiver can authenticate IPSec peers and packets sent by the IPSec sender to ensure that the data has not been altered during transmission.
- Data origin authentication—The IPSec receiver can authenticate the source of the IPSec packets sent. This service is dependent upon the data integrity service.
- Anti-replay—The IPSec receiver can detect and reject replayed packets, helping prevent spoofing and man-in-the-middle attacks.



The PIX Firewall uses the open IPSec protocol to enable secure VPNs. IPSec is a set of security protocols and algorithms used to secure data at the network layer. IPSec and related security protocols conform to open standards promulgated by the Internet Engineering Task Force (IETF) and documented RFCs and IETF-draft papers.

IPSec acts at the network layer, protecting and authenticating IP packets between a PIX Firewall and other participating IPSec devices (peers), such as PIX Firewalls, Cisco routers, the Cisco Secure VPN Client, and other IPSec-compliant products.

IPSec can be used to scale from small to very large networks. It is included in PIX Firewall version 5.0 and later.



The PIX Firewall supports the following IPSec and related standards:

- IPSec (IP Security Protocol)
- Internet Key Exchange (IKE)
- Data Encryption Standard (DES)
- Triple DES (3DES)
- Diffie-Hellman (DH)
- Message Digest 5 (MD5)
- Secure Hash Algorithm-1 (SHA-1)
- Ravist-Shamir-Adelman signatures (RSA)
- Certificate Authorities (CA)

IPSec

IPSec is a framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers at the IP layer. IPSec can be used to protect one or more data flows between IPSec peers. IPSec is documented in a series of Internet RFCs, all available at http://www.ietf.org/html.charters/ipsec-charter.html. The overall IPSec implementation is guided by "Security Architecture for the Internet Protocol," RFC 2401. IPSec consists of the following two main protocols:

- Authentication Header (AH)—A security protocol that provides authentication and optional replay-detection services. AH acts as a "digital signature" to ensure that data in the IP packet has not been tampered with. AH does not provide data encryption and decryption services. AH can be used either by itself or with Encapsulating Security Payload (ESP).

- Encapsulating Security Payload (ESP)—A security protocol that provides data confidentiality and protection with optional authentication and replay-detection services. The PIX Firewall uses ESP to encrypt the data payload of IP packets. ESP can be used either by itself or in conjunction with AH.

IKE

IKE is a hybrid protocol that provides utility services for IPSec: authentication of the IPSec peers, negotiation of IKE and IPSec security associations, and establishment of keys for encryption algorithms used by IPSec. IKE is synonymous with ISAKMP in PIX Firewall configuration.

SA

The concept of a security association (SA) is fundamental to IPSec. An SA is a connection between IPSec peers that determines the IPSec services available between the peers, similar to a TCP or UDP port. Each IPSec peer maintains an SA database in memory containing SA parameters. SAs are uniquely identified by IPSec peer address, security protocol, and security parameter index (SPI). You will need to configure SA parameters and monitor SAs on the PIX Firewall.

DES

DES is used to encrypt and decrypt packet data. DES is used by both IPSec and IKE. DES uses a 56-bit key, ensuring high performance encryption.

3DES

3DES is a variant of DES, which iterates three times with three separate keys, effectively doubling the strength of DES. 3DES is used by IPSec to encrypt and decrypt data traffic. 3DES uses a 168-bit key, ensuring strong encryption.

DH

Diffie-Hellman is a public-key cryptography protocol. It allows two parties to establish a shared secret key over an insecure communications channel. DH is used within IKE to establish session keys. 768-bit and 1024-bit DH groups are supported in the PIX Firewall. The 1024-bit group is more secure.

MD5

MD5 is a hash algorithm used to authenticate packet data. The PIX Firewall uses the MD5 hashed message authentication code (HMAC) variant which provides an additional level of hashing. A hash is a one-way encryption algorithm that takes an input message of arbitrary length and produces a fixed-length output message. IKE, AH, and ESP use MD5 for authentication.

SHA-1

SHA is a hash algorithm used to authenticate packet data. The PIX Firewall uses the SHA-1 HMAC variant which provides an additional level of hashing. IKE, AH, and ESP use SHA-1 for authentication.

RSA Signatures

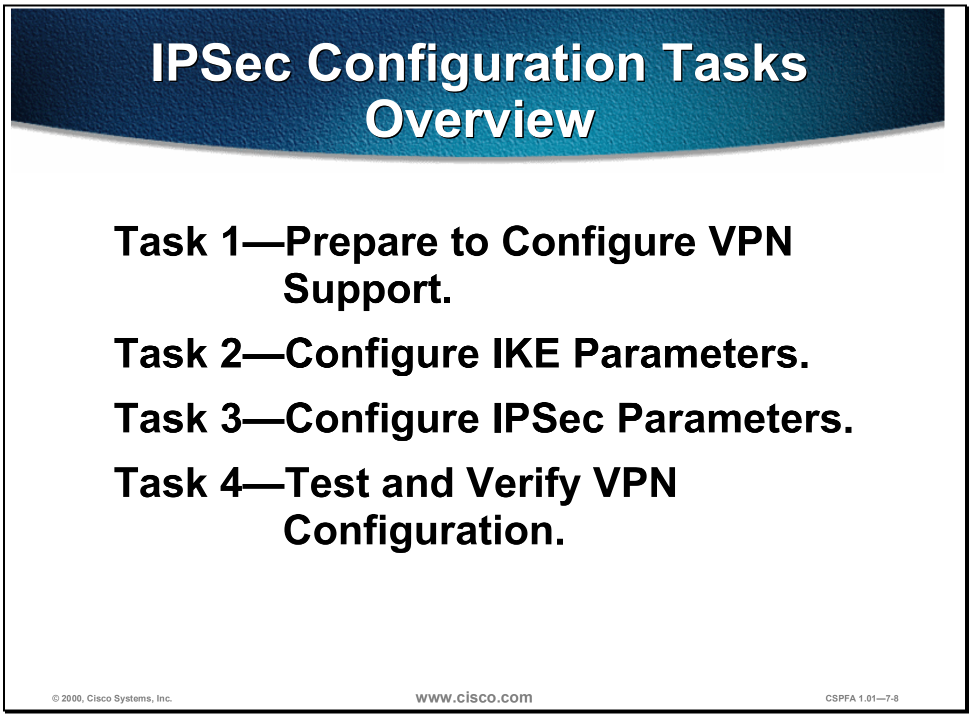
RSA is a public-key cryptographic system used for authentication. IKE on the PIX Firewall uses a DH exchange to determine secret keys on each IPSec peer used by encryption algorithms. The DH exchange can be authenticated with RSA (or pre-shared keys).

CA

The certificate authority support of the PIX Firewall allows the IPSec-protected network to scale by providing the equivalent of a digital identification card to each device. When two IPSec peers wish to communicate, they exchange digital certificates to prove their identities (thus removing the need to manually exchange public keys with each peer or to manually specify a shared key at each peer). The digital certificates are obtained from a CA. CA support on the PIX Firewall uses RSA signatures to authenticate the CA exchange.

IPSec Configuration Tasks

This section describes the tasks you perform when configuring an IPSec-based VPN.



The rest of this chapter demonstrates how to configure an IPSec-based VPN between two PIX Firewalls operating as secure gateways, using pre-shared keys for authentication. The IPSec configuration process can be summed up as two major tasks: configuring an IPSec encryption policy, and applying the policy to an interface.

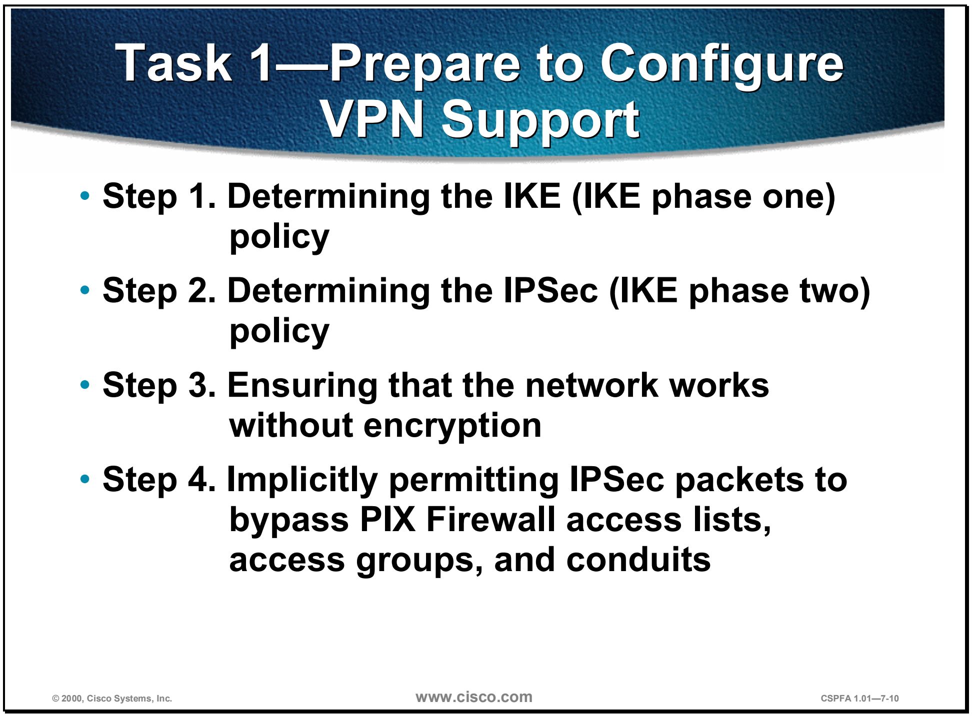
The four overall tasks used to configure IPSec encryption on the PIX Firewall are summarized below. Subsequent sections of this chapter discuss each configuration task in greater detail. The following are the four tasks:

- Task 1—Preparing to for configure VPN support. This task consists of several steps that determine IPSec policies, ensure that the network works, and ensure that the PIX Firewall can support IPSec.
- Task 2—Configuring IKE parameters. This task consists of several configuration steps that ensure that IKE can set up secure channels to desired IPSec peers. IKE can set up IPSec SAs, enabling IPSec sessions. IKE negotiates IKE parameters and sets up IKE SAs during an IKE phase one exchange called “main mode.”
- Task 3—Configure IPSec parameters. This task consists of several configuration steps that specify IPSec SA parameters between peers, and set global IPSec values. IKE negotiates SA parameters and sets up IPSec SAs during an IKE phase two exchange called “quick mode.”

- Task 4—Test and verify VPN configuration. After you configure IPSec, you will need to verify you have configured it correctly, and ensure that it works.

Task 1— Prepare to Configure VPN Support

Successful implementation of an IPSec network requires advance preparation before beginning the configuration of individual PIX Firewalls. This section outlines how to determine network design details.



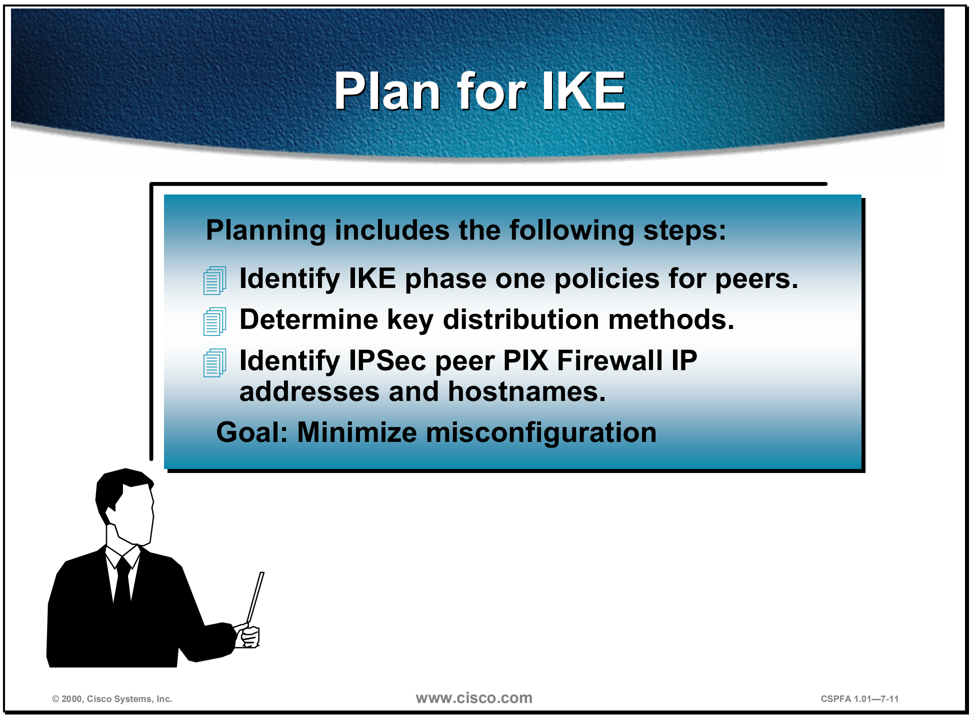
Configuring IPSec encryption can be complicated. You must plan in advance if you want to configure IPSec encryption correctly the first time and minimize misconfiguration. You should begin this task by defining the overall security needs and strategy based on the overall company security policy. Some planning steps include the following:

Step 1 Determining the IKE (IKE phase one) policy—Determine the IKE policies between peers based on the number and location of IPSec peers.

Step 2 Determining the IPSec (IKE phase two) policy—You will need to identify IPSec peer details such as IP addresses and IPSec modes. You then configure crypto maps to gather all IPSec policy details together.

Step 3 Ensuring that the network works without encryption (no excuses!)—Ensure that basic connectivity has been achieved between IPSec peers using the desired IP services before configuring PIX Firewall IPSec.

Step 4 Implicitly permitting IPSec packets to bypass PIX access lists, access groups and conduits. In this step you will must enter the **sysopt connection permit-ipsec** command.



An IKE policy defines a combination of security parameters to be used during the IKE negotiation. You should determine the IKE policy, then configure it. Some planning steps include:

- Determining IKE phase one (ISAKMP) policies for peers—An IKE policy defines a combination of security parameters to be used during the IKE negotiation. Each IKE negotiation begins by each peer agreeing on a common (shared) IKE policy. The IKE policy suites must be determined in advance of configuration.
- Determining key distribution methods based on the numbers and locations of IPSec peers—You may wish to use a CA server to support scalability of IPSec peers. You must then configure IKE to support the selected key distribution method.
- Identifying IPSec peer router IP addresses and hostnames—You will need to determine the details of all of the IPSec peers that will use IKE for establishing SAs.

The goal of advance planning is to minimize misconfiguration.

IKE Phase One Policy Parameters

Parameter	Strong	Stronger
Encryption Algorithm	DES	3DES
Hash Algorithm	MD5	SHA-1
Authentication Method	Pre-share	RSA Signature
Key Exchange	DH Group 1	DH Group 2
IKE SA Lifetime	86,400 seconds	< 86,400 seconds

 www.cisco.com CSPFA 1.01—7-12

An IKE policy defines a combination of security parameters to be used during the IKE negotiation. A group of policies makes up a "protection suite" of multiple policies that enable IPSec peers to establish IKE sessions and SAs with a minimum of configuration.

Create IKE Policies for a Purpose

IKE negotiations must be protected, so each IKE negotiation begins by each peer agreeing on a common (shared) IKE policy. This policy states which security parameters will be used to protect subsequent IKE negotiations.

After the two peers agree upon a policy, the security parameters of the policy are identified by a security association established at each peer, and these security associations apply to all subsequent IKE traffic during the negotiation.

You can create multiple, prioritized policies at each peer to ensure that at least one policy will match a remote peer's policy.

Define IKE Policy Parameters

You can select specific values for each IKE parameter, per the IKE standard. You choose one value over another based on the security level you desire and the type of IPSec peer to which you will connect.

There are five parameters to define in each IKE policy, as outlined in the figure and in the following table. The figure shows the relative strength of each parameter, and the table shows the default values.

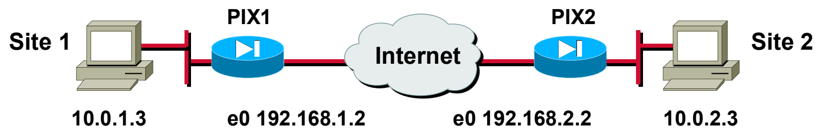
IKE Policy Parameters

Parameter	**Accepted Values**	**Keyword**	**Default**
Message encryption algorithm	56-bit DES 168-bit 3DES	**des** **3des**	DES
Message integrity (hash) algorithm	SHA-1 (HMAC variant) MD5 (HMAC variant)	**sha** **md5**	SHA-1
Peer authentication method	Pre-shared keys RSA signatures	**pre-share** **rsa-sig**	RSA signatures
Key exchange parameters (Diffie-Hellman group identifier)	768-bit Diffie-Hellman or 1024-bit Diffie-Hellman	**1** **2**	768-bit Diffie-Hellman
ISAKMP-established security association's lifetime	Can specify any number of seconds	—	86,400 seconds (1 day)

Note 3DES provides stronger encryption than DES. Some tradeoffs of 3DES are that it takes more processing power, and it may be restricted for export or import into some countries.

Note RSA signatures are used with CA support, and require enrollment to a CA server.

Determine IKE Phase One Policy

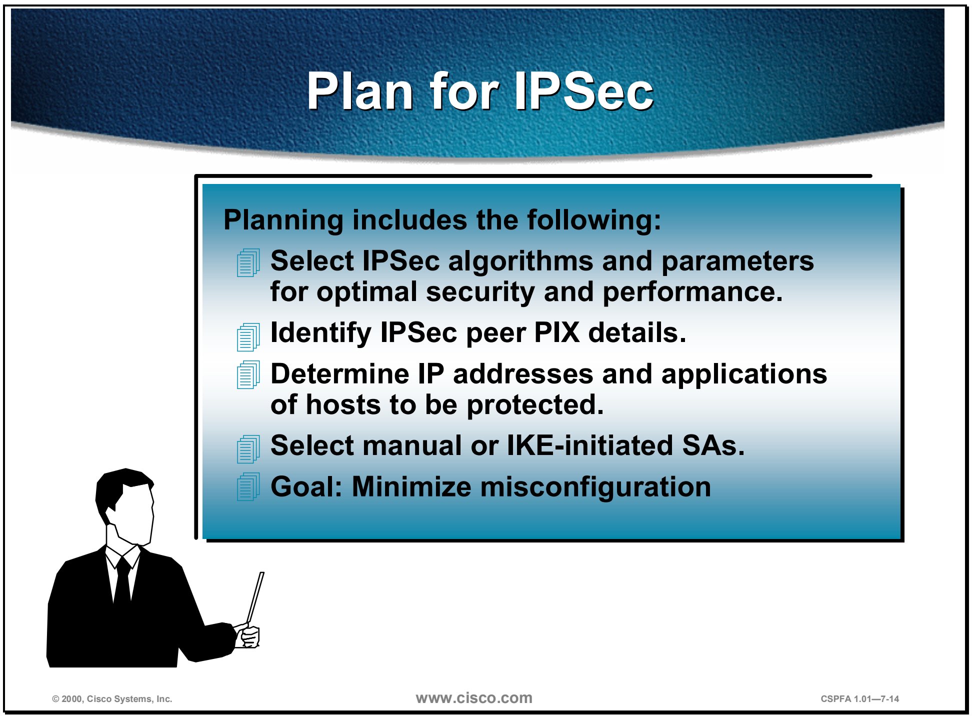


Parameter	Site 1	Site 2
Encryption Algorithm	DES	DES
Hash Algorithm	SHA	SHA
Authentication Method	Pre-share	Pre-share
Key Exchange	768-bit D-H	768-bit D-H
IKE SA Lifetime	86,400 seconds	86,400 seconds

 www.cisco.com CSPFA 1.01—7-13

An IKE policy defines a combination of security parameters to be used during the IKE negotiation. A group of policies makes up a “protection suite” of multiple policies that enable IPSec peers to establish IKE sessions and SAs with a minimum of configuration.

You should determine IKE policy details for each IPSec peer before configuring IKE. The figure shows a summary of some IKE policy details that will be configured in the examples in this chapter.

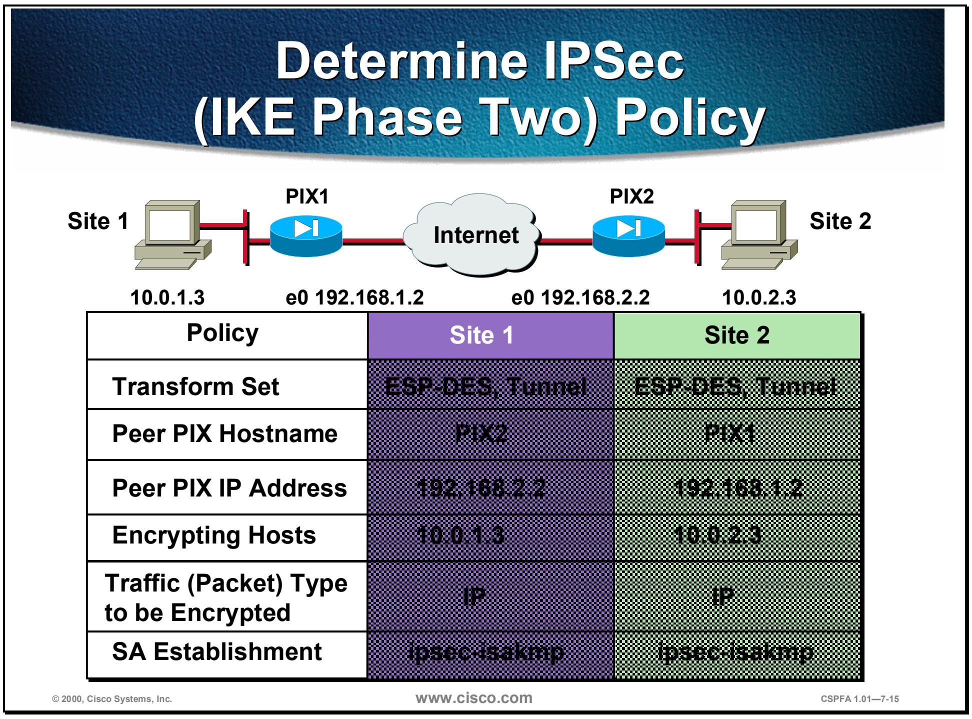


Planning for IPSec (IKE phase two) is another important step you should complete before actually configuring the PIX Firewall. Items to determine at this stage include the following:

- Select IPSec algorithms and parameters for optimal security and performance. You should determine what type of IPSec security will be used to secure interesting traffic. Some IPSec parameters require you to make tradeoffs between high performance and stronger security.
- Identify IPSec peer details. You must identify the IP addresses and hostnames of all IPSec peers you will connect to.
- Determine IP addresses and applications of hosts to be protected at the local peer and remote peer.
- Decide whether security associations are manually established or are established via IKE.

Note IPSec security associations can be configured manually, but is not recommended because IKE is easier to configure.

The goal of this planning step is to gather the precise data you will need in later steps to minimize misconfiguration.



Determining network design details includes defining a more detailed security policy for protecting traffic. You can then use the detailed policy to help select IPSec transform sets and modes of operation. Your security policy should answer the following questions:

- What protections are required or are acceptable for the protected traffic?
- What traffic should or should not be protected?
- Which PIX interfaces are involved in protecting internal nets, external nets, or both?
- What are the peer IPSec endpoints for the traffic?
- How should SAs be established?

The figure above shows a summary of IPSec encryption policy details that will be configured in the examples in this chapter.

Task 2— Configure IKE Parameters

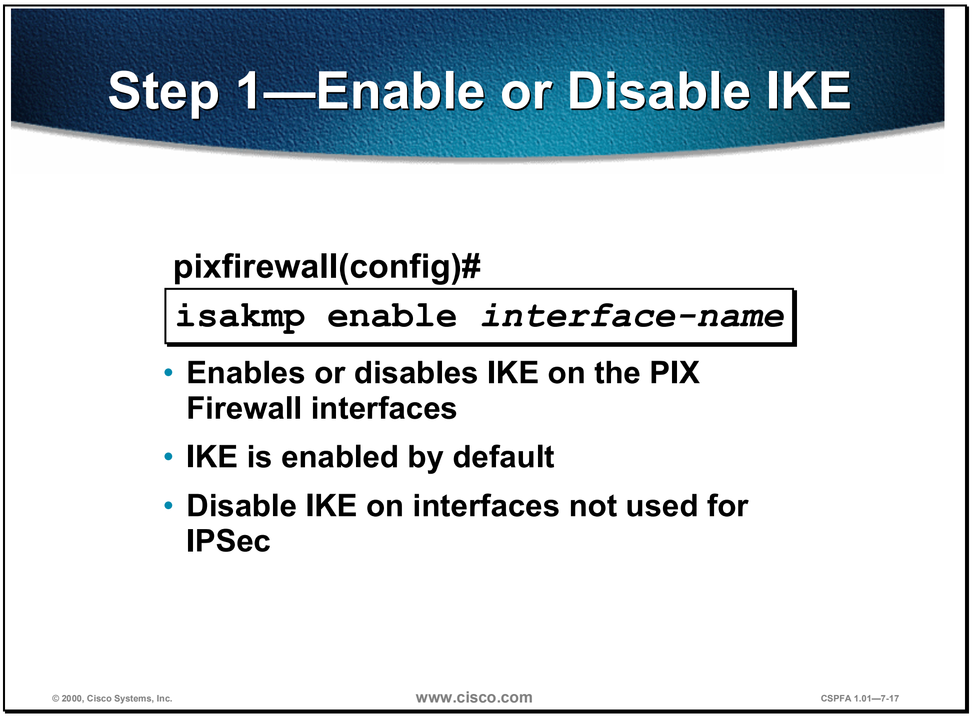
The next major task in configuring PIX Firewall IPSec is to configure IKE parameters gathered in the previous task. This section presents the steps used to configure IKE parameters for IKE pre shared keys:

Step 1 Enable or disable IKE.

Step 2 Configure an IKE phase one policy.

Step 3 Configure the IKE pre-shared key.

Step 4 Verify IKE phase one details.

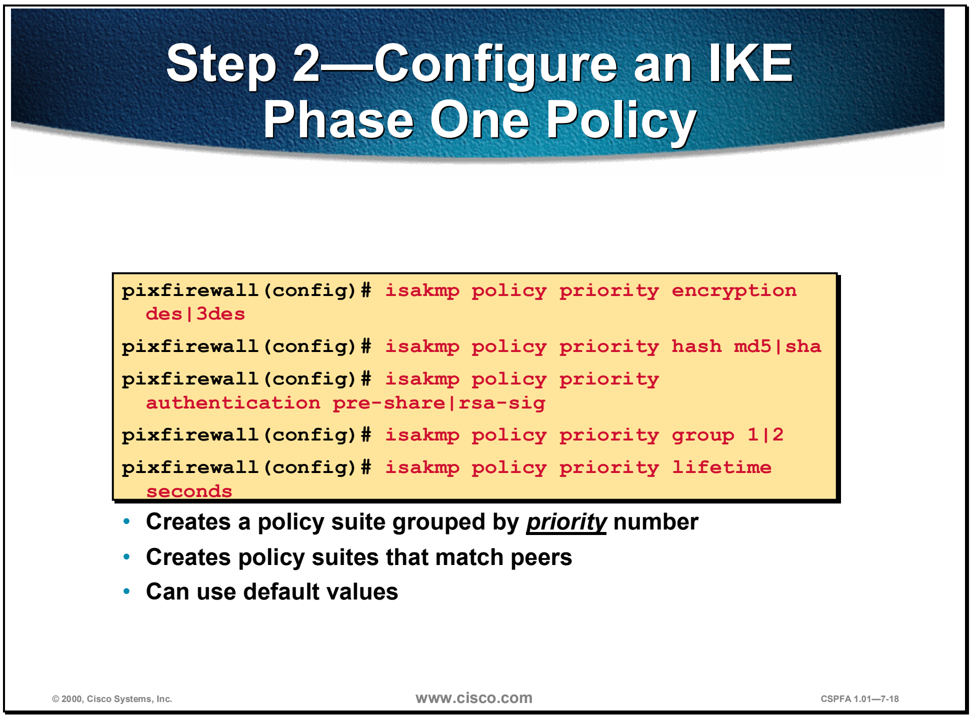


Step 1 Enable or disable IKE (ISAKMP) negotiation:

pixfirewall(config)# **isakmp enable** ***interface-name***

Specify the PIX Firewall interface on which the IPSec peer will communicate. IKE is enabled by default and for individual PIX Firewall interfaces. Use the **no isakmp enable** ***interface-name*** command to disable IKE.

Note PIX Firewall version 5.0 software supports IPSec termination on the outside interface only.



Step 2 Configure an IKE Phase one policy with the **isakmp policy** command to match expected IPSec peers:

(a) Identify the policy with a unique priority number:

pixfirewall(config)# **isakmp policy *priority***

(b) Specify the encryption algorithm. The default is **des**:

pixfirewall(config)# **isakmp policy *priority* encryption des | 3des**

(c) Specify the hash algorithm. The default is **sha**:

pixfirewall(config)# **isakmp policy *priority* hash md5 | sha**

(d) Specify the authentication method:

pixfirewall(config)# **isakmp policy *priority* authentication pre-share | rsa-sig**

Note If you specify the authentication method of pre-shared keys, you must manually configure these keys, which is outlined in Step 3.

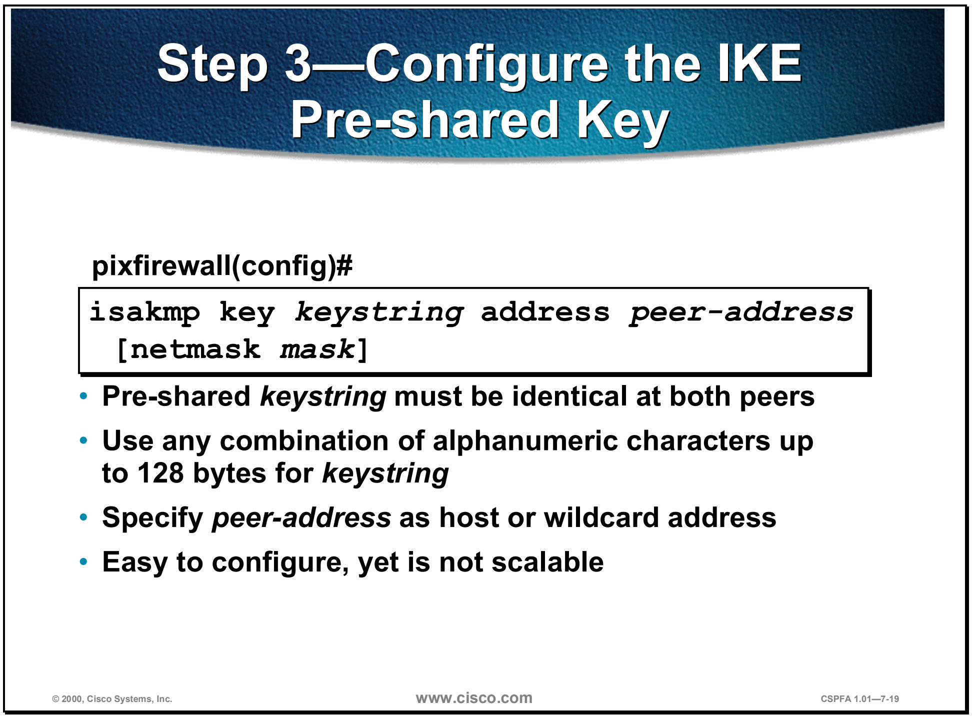
(e) Specify the Diffie-Hellman group identifier. The default is **group 1**:

pixfirewall(config)# **isakmp policy *priority* group 1|2**

(f) Specify the IKE security association's lifetime. The default is ***86400***.

pixfirewall(config)# **isakmp policy *priority* lifetime *seconds***

Note PIX Firewall software has preset default values. If you enter a default value for a given policy parameter, it will not be written in the configuration. If you do not specify a value for a given policy parameter, the default value is assigned. You can observe configured and default values with the **show isakmp policy** command.



Step 3 Configure the IKE pre-shared key.

pixfirewall(config)# **isakmp key *keystring* address *peer-address* [*netmask*]**

The ***keystring*** is any combination of alphanumeric characters up to 128 bytes. This pre-shared key must be identical at both peers.

The ***peer-address*** and ***netmask*** should point to the IP address of the IPSec peer. A wildcard peer address and netmask of **0.0.0.0 0.0.0.0** may be configured to share the pre-shared key among many peers. However, Cisco strongly recommends using a unique key for each peer.

You can also use the peer's hostname for the pre-shared key.

Step 4—Verify IKE Phase One Policies

```
pixfirewall# show isakmp policy
Protection suite of priority 10
        encryption algorithm:   DES - Data Encryption Standard (56 bit keys).
        hash algorithm:         Secure Hash Standard
        authentication method:  Pre-Shared Key
        Diffie-Hellman group:   #1 (768 bit)
        lifetime:               86400 seconds, no volume limit
Default protection suite
        encryption algorithm:   DES - Data Encryption Standard (56 bit keys).
        hash algorithm:         Secure Hash Standard
        authentication method:  Rivest-Shamir-Adleman Signature
        Diffie-Hellman group:   #1 (768 bit)
        lifetime:               86400 seconds, no volume limit
```

- **Displays configured and default IKE protection suites**

 www.cisco.com CSPFA 1.01—7-20

Step 4 Verify IKE phase one policies.

The **show isakmp policy** command displays configured and default policies, as shown in the figure. The **show isakmp** command displays configured policies much as they would appear with the **write terminal** command, as follows:

```
pix1(config)# show isakmp
isakmp enable outside
isakmp key cisco123 address 192.168.2.2 netmask 255.255.255.255
isakmp policy 10 authentication pre-share
isakmp policy 10 encryption des
isakmp policy 10 hash sha
isakmp policy 10 group 1
isakmp policy 10 lifetime 86400
```

Task 3—Configure IPSec Parameters

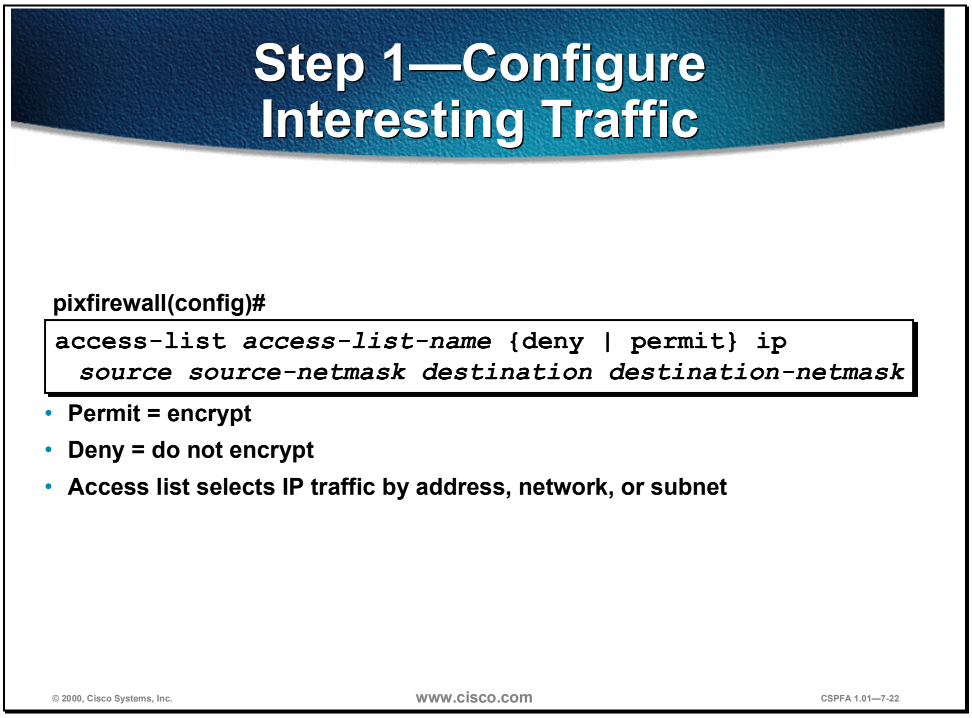
The next major task in configuring PIX Firewall IPSec is to configure the previously gathered IPSec parameters. This section presents the steps used to configure IPSec parameters for IKE pre-shared keys:

Step 1 Configure interesting traffic.

Step 2 Configure a transform set.

Step 3 Configure the crypto map.

Step 4 Apply the crypto map to the interface.



Step 1 Configure interesting traffic with crypto access lists:

`pixfirewall(config)#` **`access-list`** ***`access-list-name`*** **`{deny | permit}`** ***`protocol source source-netmask destination destination-netmask`***

- **permit** causes all IP traffic that matches the specified conditions to be protected by crypto, using the policy described by the corresponding crypto map entry.
- **deny** instructs the PIX Firewall to route traffic in the clear.
- ***source*** and ***destination*** are networks, subnets, or hosts.
- ***protocol*** indicates which IP packet types to encrypt.

Note PIX Firewall version 5.0 supports the IP protocol only. PIX Firewall version 5.1 supports greater protocol and port granularity.

Crypto access lists are traffic selection access lists. They are used to define which IP traffic is interesting and will be protected by IPSec, and which traffic will not be protected by IPSec. Crypto access lists perform the following functions:

- Indicate the data flow to be protected by IPSec
- Select outbound traffic to be protected by IPSec
- Process inbound traffic in order to filter out and discard traffic that should be protected by IPSec
- Determine whether or not to accept requests for IPSec security associations for the requested data flows when processing IKE negotiations

---

Note Although the access list syntax is unchanged from access lists applied to PIX Firewall interfaces, the meanings are slightly different for crypto access lists—**permit** specifies that matching packets must be encrypted while **deny** specifies that matching packets need not be encrypted.

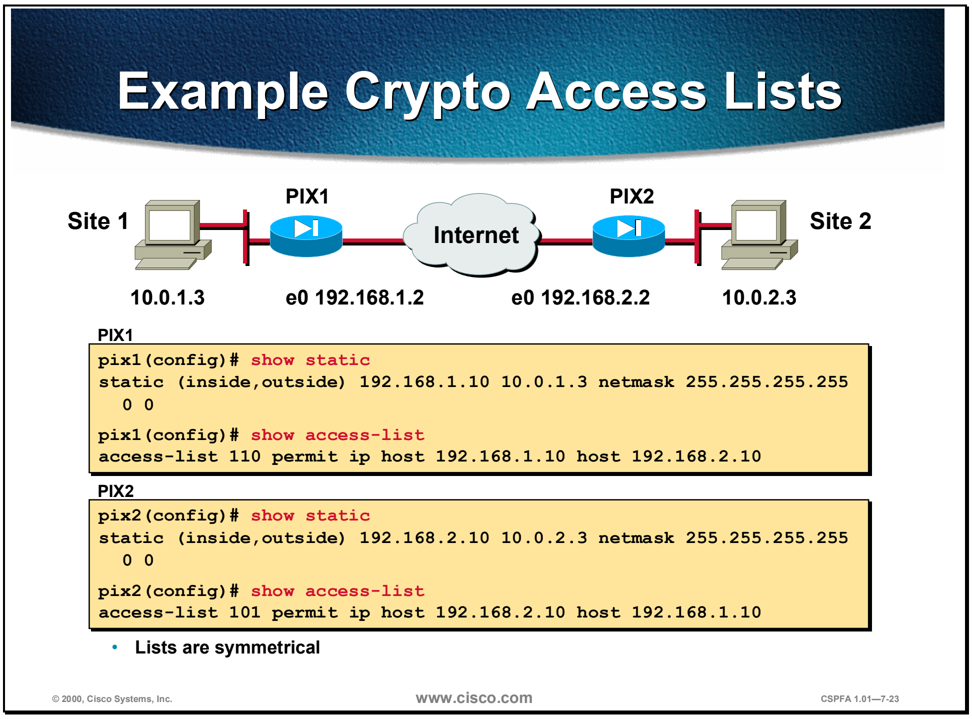
Any unprotected inbound traffic that matches a permit entry in the crypto access list for a crypto map entry, flagged as IPSec, will be dropped, since this traffic was expected to be protected by IPSec.

If you want certain traffic to receive one combination of IPSec protection (for example, authentication only) and other traffic to receive a different combination of IPSec protection (for example, both authentication and encryption), you must create two different crypto access lists to define the two different types of traffic. These different access lists are then used in different crypto map entries which specify different IPSec policies.

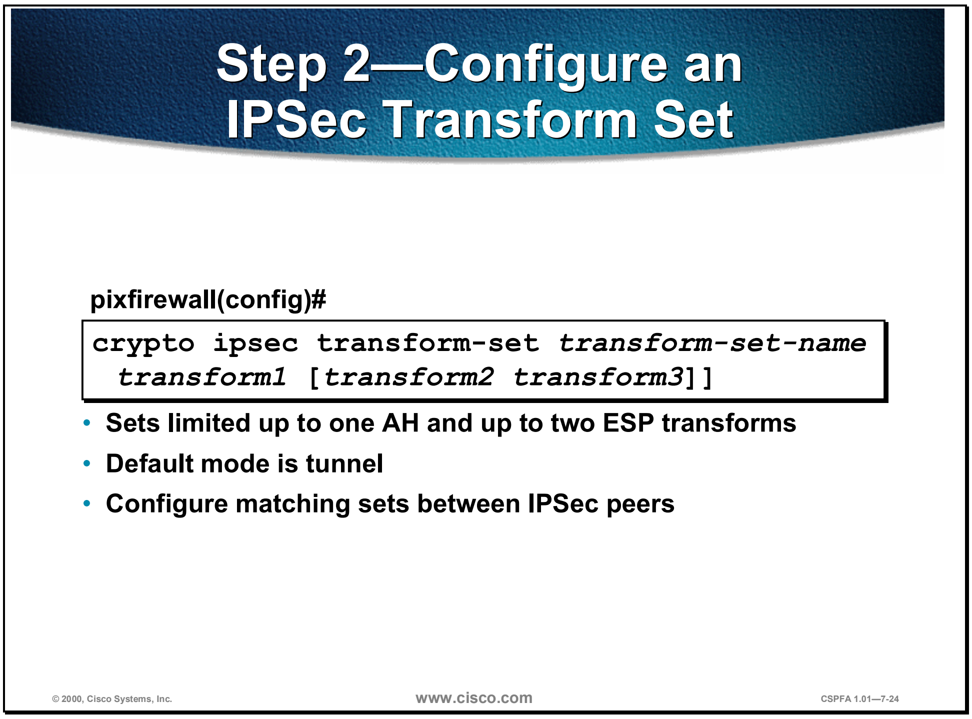
In a later configuration step, you will associate the crypto access lists to particular interfaces when you configure and apply crypto map sets to the interfaces.

WARNING *Cisco recommends that you avoid using the* ***any*** *keyword to specify source or destination addresses. The* ***permit any any*** *statement is strongly discouraged, as this will cause all outbound traffic to be protected (and all protected traffic sent to the peer, specified in the corresponding crypto map entry), and will require protection for all inbound traffic. Then, all inbound packets that lack IPSec protection will be silently dropped, including packets for routing protocols, NTP, echo, echo response, and so on.*

Try to be as restrictive as possible when defining which packets to protect in a crypto access list. If you must use the **any** keyword in a permit statement, you must preface that statement with a series of deny statements to filter out any traffic (that would otherwise fall within that permit statement) that you do not want protected.



Use the **show access-list** command to display currently configured access lists. The figure above contains an example access list for each of the peer PIX Firewalls. Each PIX Firewall in this example has static mapping of a global IP address to an inside host. The access list ***source*** field is configured for the global IP address of the local PIX Firewall's static, which is the ***destination*** field for the peer PIX Firewall's global IP address. The access lists are symmetrical.



Step 2 Configure an IPSec transform set:

pixfirewall(config)# **crypto ipsec transform-set** ***transform-set-name transform1*** **[*transform2 transform3*]]**

- ***transform-set-name***—The name of the transform set to create or modify.
- ***transform1*** **[*transform2 transform3*]]**—Specify up to three transforms.

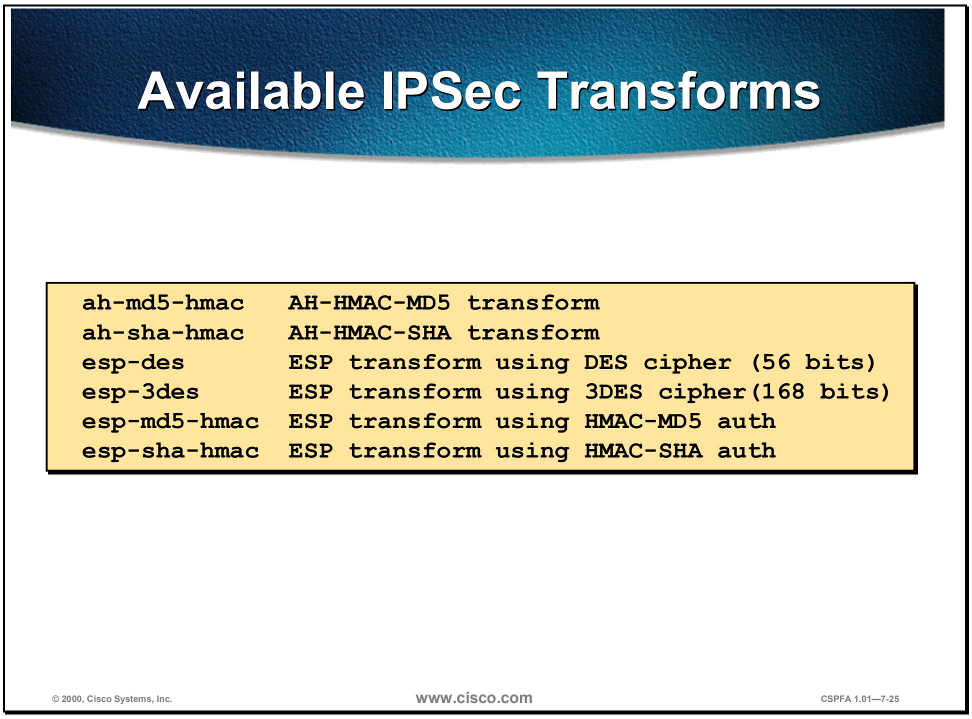
- Sets are limited to up to one AH and up to two ESP transforms.
- The default mode is tunnel.
- Configure matching transform sets between IPSec peers.

Transforms define the IPSec security protocols and algorithms. Each transform represents an IPSec security protocol (ESP, AH, or both) plus the algorithm you want to use.

You can specify multiple transform sets, and then specify one or more of these transform sets in a crypto map entry. The transform set defined in the crypto map entry will be used in the IPSec security association negotiation to protect the data flows specified by the access list of that crypto map entry.

During the IPSec security association negotiation, the peers agree to use a particular transform set for protecting a particular data flow.

A transform set equals an AH transform and an ESP transform plus the mode (transport or tunnel.)



The PIX Firewall supports the transform sets listed in the figure.

Choosing IPSec transforms combinations can be complex. The following tips may help you select transforms that are appropriate for your situation.

If you want to provide data confidentiality, include an ESP encryption transform.

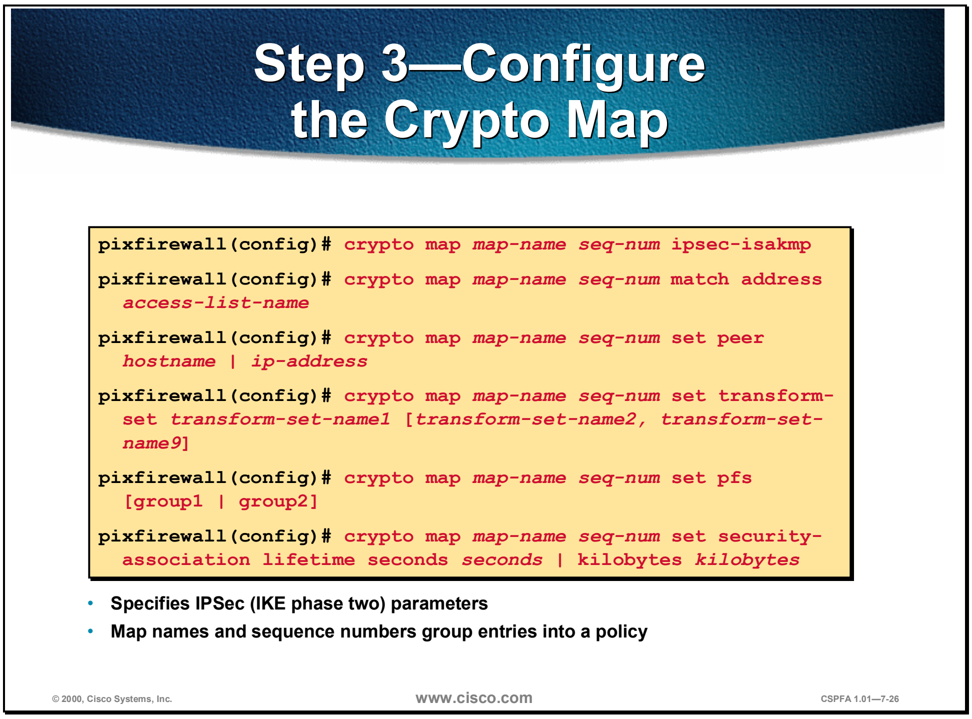
Also consider including an ESP authentication transform or an AH transform to provide authentication services for the transform set:

- Ensure data authentication for the outer IP header as well as the data, include an AH transform.
- Ensure data authentication (using either ESP or AH), you can choose from the MD5 or SHA (HMAC keyed hash variants) authentication algorithms.

The SHA algorithm is generally considered stronger than MD5, but it is slower.

Examples of acceptable transform combinations are as follows:

- **esp-des** for high performance encryption
- **ah-md5-hmac** for authenticating packet contents with no encryption
- **esp-3des** and **esp-md5-hmac** for strong encryption and authentication
- **ah-sha-hmac** and **esp-3des** and **esp-sha-hmac** for strong encryption and authentication



Step 3 Configure the crypto map with the **crypto map** command.

(a) Create a crypto map entry in IPSec ISAKMP mode:

pixfirewall(config)# **crypto map *map-name seq-num* ipsec-isakmp**

- This identifies the crypto map with a unique crypto map name and sequence number.

(b) Assign an access list to the crypto map entry:

pixfirewall(config)# **crypto map *map-name seq-num* match address *access-list-name***

(c) Specify the peer to which the IPSec protected traffic can be forwarded:

pixfirewall(config)# **crypto map *map-name seq-num* set peer *hostname* | *ip-address***

- Set the peer hostname or IP address.
- Specify multiple peers by repeating this command.

(d) Specify which transform sets are allowed for this crypto map entry.

pixfirewall(config)# **crypto map *map-name seq-num* set transform-set *transform-set-name1* [*transform-set-name2, transform-set-name9*]**

- List multiple transform sets in order of priority (highest priority first).
- You can specify up to nine (9) transform sets.

(e) (Optional) Specify whether IPSec should ask for perfect forward secrecy (PFS) when requesting new security associations for this crypto map entry, or should require PFS in requests received from the peer:

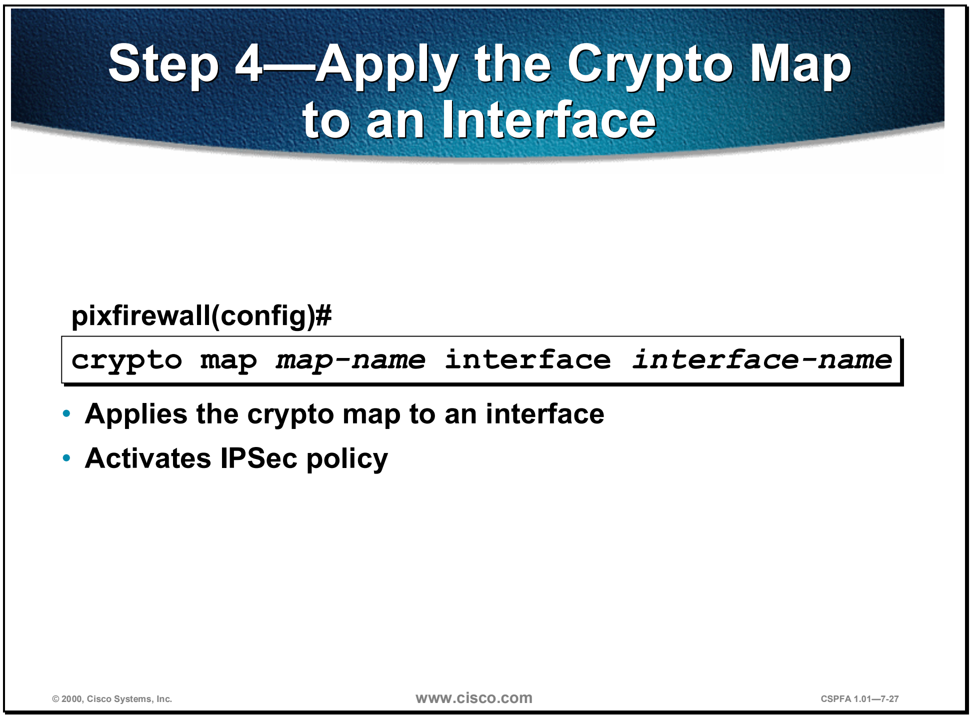
pixfirewall(config)# **crypto map *map-name seq-num* set pfs [group1 | group2]**

Note PFS p rovides additional security for Diffie-Hellman key exchanges at a cost of additional processing.

(f) (Optional) Specify the security association lifetime for the crypto map entry if you want the security associations for this entry to be negotiated using different IPSec security association lifetimes other than the global lifetimes:

pixfirewall(config)# **crypto map** ***map-name seq-num*** **set security-association lifetime seconds** ***seconds*** **| kilobytes** ***kilobytes***

(g) (Optional) Specify dynamic crypto maps with the **crypto dynamic-map** ***dynamic-map-name dynamic-seq-num*** command. A dynamic crypto map entry is essentially a crypto map entry without all the parameters configured. It acts as a policy template where the missing parameters are later dynamically configured (as the result of an IPSec negotiation) to match a peer's requirements. This allows peers to exchange IPSec traffic with the PIX Firewall even if the PIX Firewall does not have a crypto map entry specifically configured to meet all the peer's requirements.

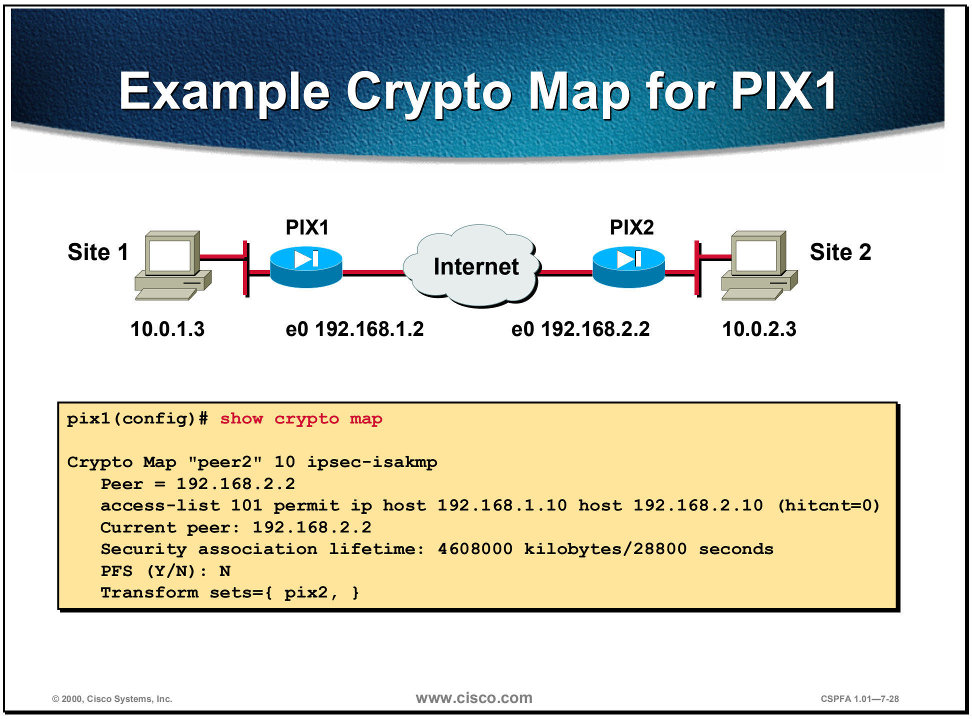


Step 4 Apply the crypto map to an interface:

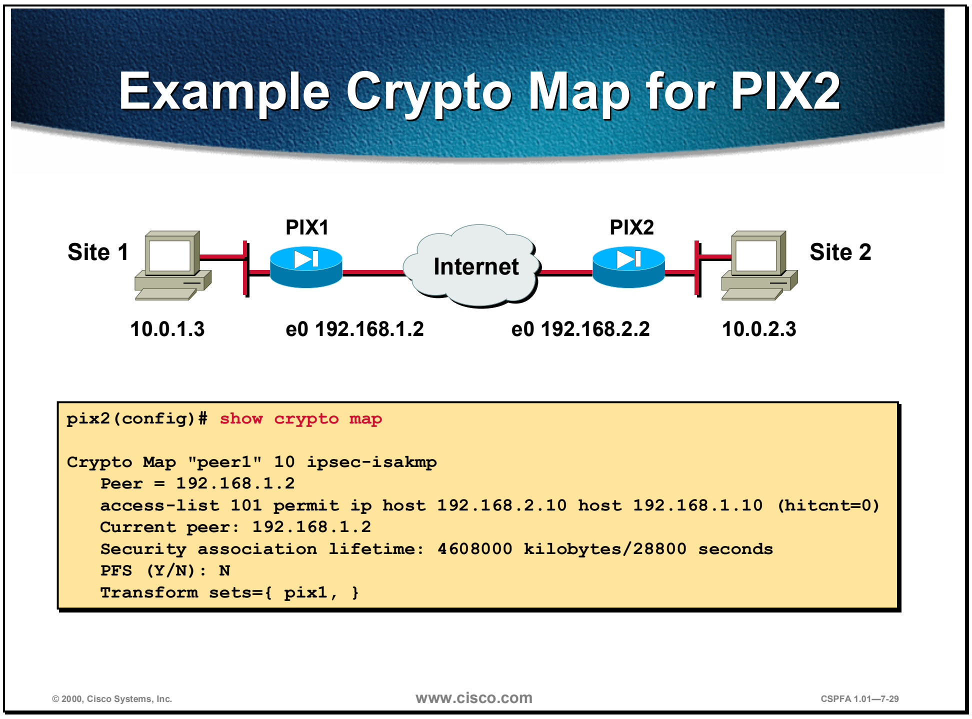
`pixfirewall(config)#` **`crypto map`** ***`map-name`*** **`interface`** ***`interface-name`***

This command applies the crypto map to an interface and the command activates the IPSec policy.

Note PIX Firewall version 5.0 supports application of IPSec encryption on the outside interface only.



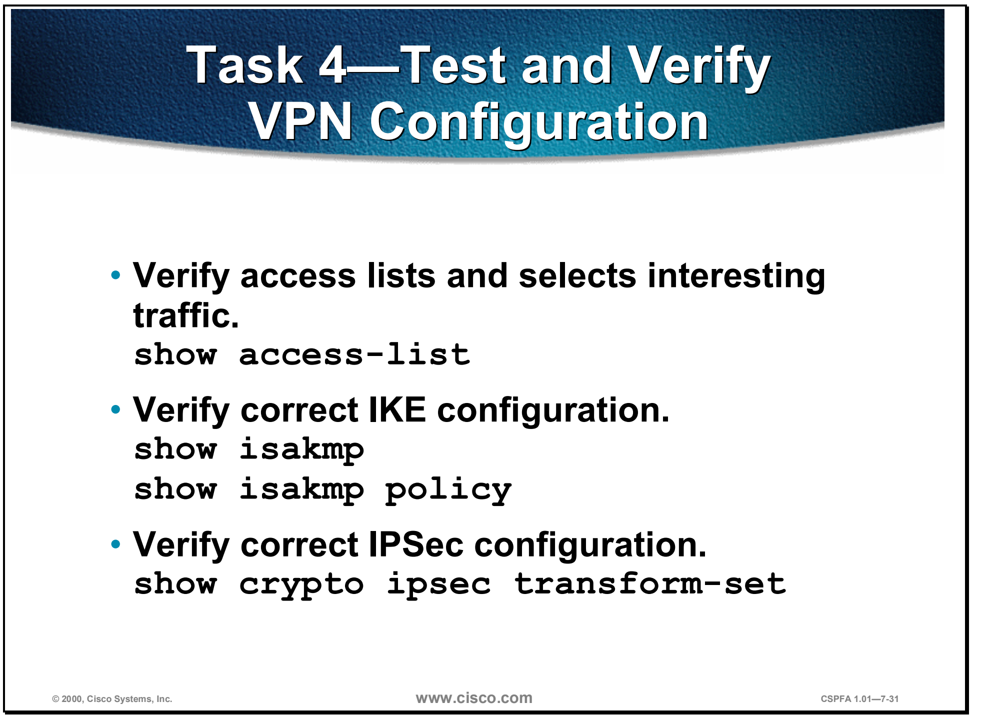
Use the **show crypto map** command to verify the crypto map configuration. Consider the example of a crypto map for PIX1 in the figure.



Consider the example of a crypto map for PIX2 in the figure.

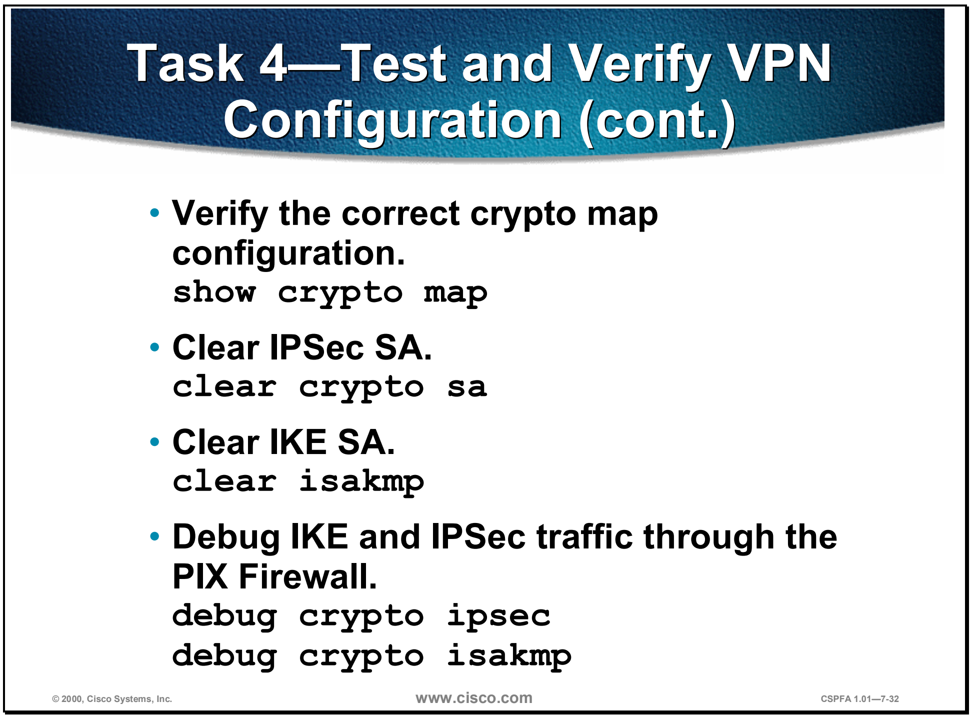
Task 4—Test and Verify VPN Configuration

The last major task in configuring PIX Firewall IPSec is to test and verify the IKE and IPSec configurations accomplished in the previous tasks. This section presents the methods and commands used to test and verify VPN configuration.



You can perform the following actions to test and verify that you have correctly configured VPN on the PIX Firewall:

- Verify access lists and selects interesting traffic with the **show access-list** command.
- Verify correct IKE configuration with the **show isakmp** and **show isakmp policy** commands.
- Verify correct IPSec configuration of transform sets with **the show crypto ipsec transform-set** command.

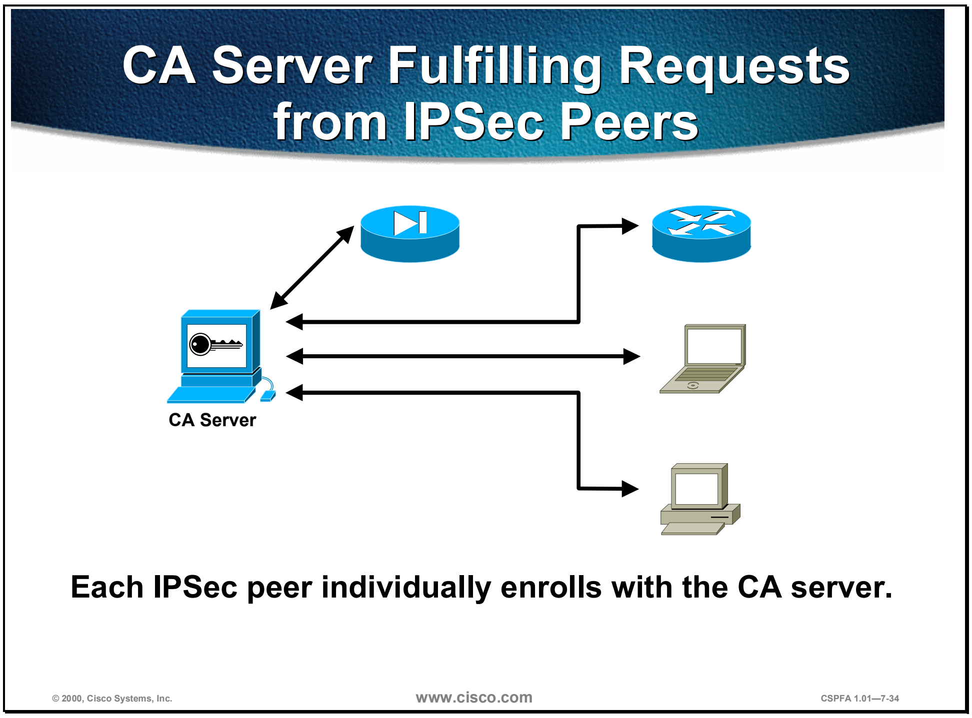


You can perform the following actions to test and verify that you have correctly configured VPN on the PIX Firewall:

- Verify the correct crypto map configuration with the **show crypto map** command.
- Clear IPSec SAs for testing of SA establishment with the **clear crypto sa** command.
- Clear IKE SAs for testing of IKE SA establishment with the **clear isakmp** command.
- Debug IKE and IPSec traffic through the PIX Firewall with the **debug crypto ipsec** and **debug crypto isakmp** commands.

Scale PIX Firewall VPNs

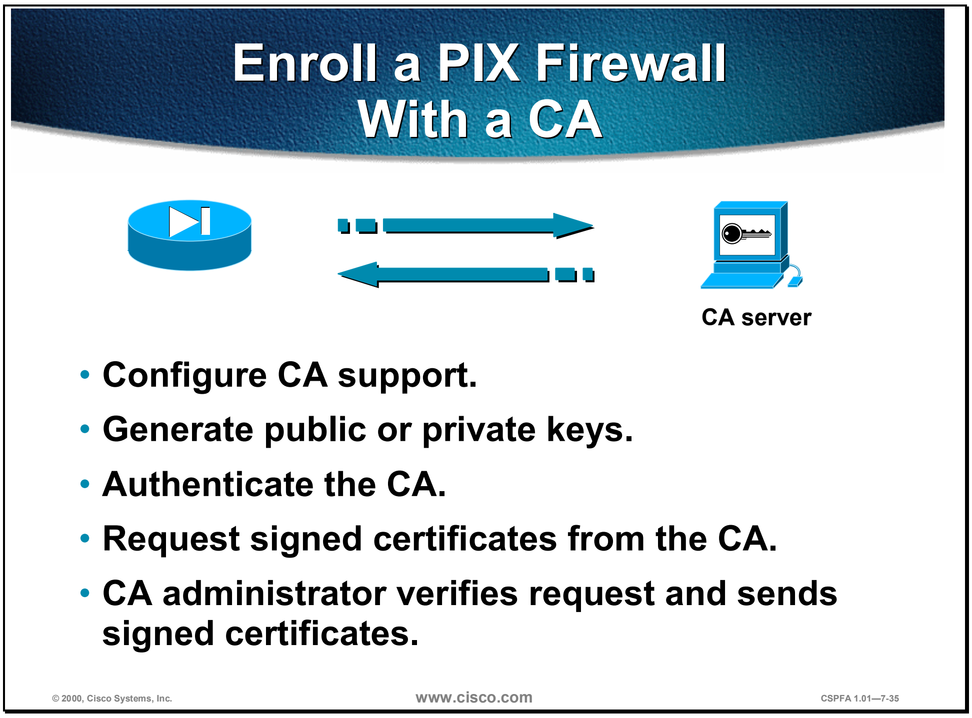
The use of pre-shared keys for IKE authentication only works when you have a few IPSec peers. Certificate Authorities enable scaling to a large number of IPSec peers.



Using a CA server is the most scalable solution. Other IKE authentication methods require manual intervention to generate and distribute the keys on a per-peer basis. The CA server enrollment process can be largely automated so that it scales well to large deployments. Each IPSec peer individually enrolls with the CA server and obtains public and private encryption keys compatible with other peers enrolled with the server.

PIX Firewall with CA Enrollment

The following section describes how to utilize a Certificate Authority to enroll a PIX Firewall.



Peers enroll with a CA server in a series of steps in which specific keys are generated and then exchanged by the PIX Firewall and the CA server to ultimately form a signed certificate. The enrollment steps can be summarized as follows:

Step 1 The PIX Firewall generates an RSA key pair.

Step 2 The PIX Firewall obtains a public key and its certificate from the CA server.

Step 3 The PIX Firewall requests signed certificate from the CA using the generated RSA keys and the public key/certificate from the CA server.

Step 4 The CA administrator verifies the request and sends a signed certificate.

Note See the "About CA" and "Configuring CA" sections in the "Configuring IPSec" chapter of the Configuration Guide for the Cisco Secure PIX Firewall for more details on how CA servers work and how to configure the PIX Firewall for CA support.

Lab Exercise: Configure a PIX Firewall VPN

Complete the following laboratory exercise to practice what you learned in this chapter.

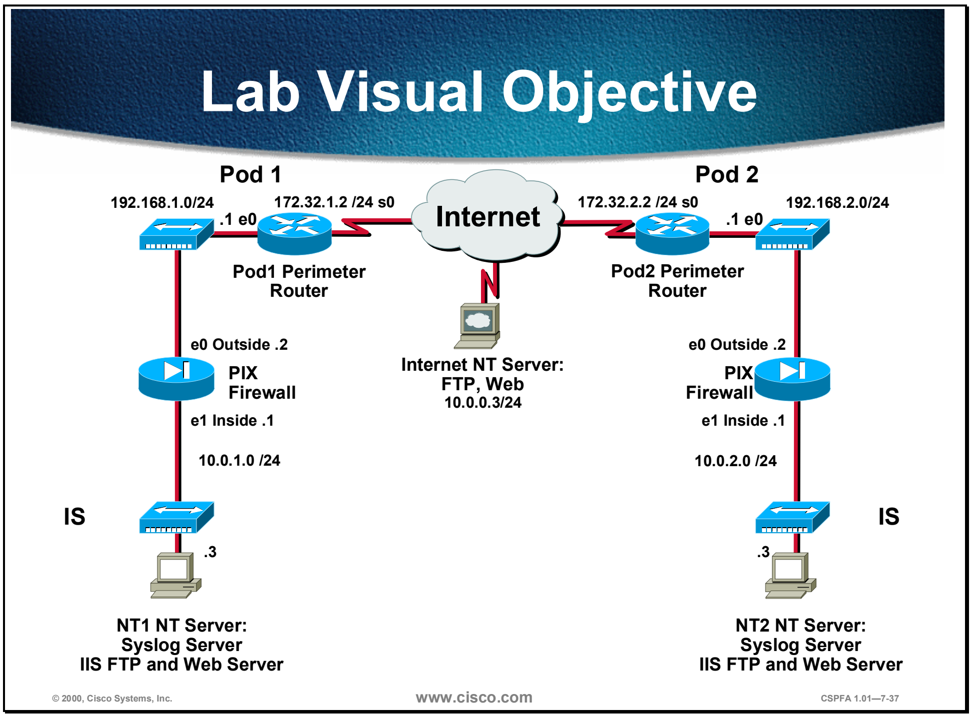
Objectives

In this lab you will complete the following tasks:

- Configure IPSec between two PIX Firewalls using IKE pre-shared keys.
- Test and verify IPSec configuration.

Visual Objective

The following figure displays the configuration you will complete in this laboratory exercise.



Scenario

The XYZ Company has purchased Cisco Secure PIX Firewalls to create a secure VPN over the Internet between sites. The company wants you to configure a secure VPN gateway using IPSec between two PIX Firewalls.

Setup

Before starting this lab, set up your equipment as follows:

- Ensure your Windows NT server is turned on
- Access the PIX Firewall console port. You may wish to save the PIX Firewall configuration to a text file for later analysis.
- Make sure the PIX Firewall is turned on.
- Ensure you can ping from your internal Windows NT server to the opposite pod group's Windows NT server.
- Ensure the Web server is running on your own internal Windows NT server.
- Ensure you can establish a Web connection from a Web browser on your internal Windows NT server to the opposite pod group's Windows NT server.

Directions

Your task in this exercise is to configure the PIX Firewall to enable IPSec encrypted tunnels to another PIX Firewall. Work with your lab partner to perform the following steps in this lab:

- Prepare to configure VPN support.
- Configure IKE parameters.
- Configure IPSec parameters.
- Test and verify IPSec configuration.

Task 1: Prepare to Configure VPN Support

Perform the following lab steps to prepare for the IKE and IPSec configuration.

Step 1 Determine the IKE and IPSec policy. In this exercise, you will use default values except when you are directed to enter a specific value.

- IKE policy is to use pre-shared keys.
- IPSec policy is to use ESP mode with DES encryption.

Step 2 Verify that a static translation is configured from a global IP address on the outside interface to the internal Windows NT server:

```
pixP(config)# show static
```

```
static (inside,outside) 192.168.P.10 10.0.P.3 netmask 255.255.255.255 0 0
```

(where *P* = pod number)

Step 3 Verify a conduit permitting Web access to your internal Windows NT server has been configured.

```
pixP(config)# show conduit
conduit permit tcp host 192.168.P.10 eq www any
```

(where *P* = pod number)

Step 4 Ensure you can establish a Web connection between pods from the Windows NT server using the static and conduit.

Step 5 Enable the PIX Firewall to implicitly permit any packet from an IPSec tunnel and bypass the checking with an associated conduit or access-group command for IPSec connections.

```
pixP(config)# sysopt connection permit-ipsec
```

Task 2: Configure IKE Parameters

Perform the following steps to configure IKE on your PIX Firewall:

Step 1 Ensure IKE is enabled on the outside interface:

```
pixP(config)# isakmp enable outside
```

Step 2 Configure a basic IKE policy using pre-shared keys for authentication.

```
pixP(config)# isakmp policy 10 authentication pre-share
```

Step 3 Set the IKE identity.

```
pixP(config)# isakmp identity address
```

Step 4 Configure the ISAKMP pre-shared key to point to the outside IP address of the peer PIX Firewall.

```
pixP(config)# isakmp key cisco123 address 192.168.Q.2 netmask 255.255.255.255
```

(where *P* = pod number, and *Q* = peer pod number)

Task 3: Configure IPSec Parameters

Perform the following steps to configure IPSec (IKE Phase two) parameters:

Step 1 Create an access list to select traffic to protect. The access list should protect IP traffic between Windows NT servers on the peer PIX Firewalls:

```
pixP(config)# access-list 101 permit ip host 192.168.P.10 host 192.168.Q.10
```

(where *P* = pod number, and *Q* = peer pod number)

Consider the example access list for PIX1 peering to PIX2:

```
pix1(config)# show access-list
access-list 101 permit ip host 192.168.1.10 host 192.168.2.10
```

Step 2 Configure an IPSec transform set (IKE phase two parameters) to use ESP and DES. Use a *transform-set-name* of pixQ, where “Q” equals your peer’s pod number:

```
pixP(config)# crypto ipsec transform-set pixQ esp-des
```

Step 3 Create a crypto map by performing the following sub-steps:

1. Create a crypto map entry. Use a *map-name* of peerQ:

```
pixP(config)# crypto map peerQ 10 ipsec-isakmp
```

(where *Q* = peer pod number)

2. Look at the crypto map and observe the defaults:

```
pixP(config)# show crypto map
Crypto Map "peerQ" 10 ipsec-isakmp
        No matching address list set.
        Current peer: 0.0.0.0
        Security association lifetime: 4608000 kilobytes/28800 seconds
        PFS (Y/N): N
        Transform sets={ }
```

What is the default security association lifetime?
A: 4608000 kilobytes/28800 seconds

---

3. Assign the access list to the crypto map.

```
pixP(config)# crypto map peerQ 10 match address 101
```

(where *Q* = peer pod number)

4. Define the peer. The peer IP address should be set to the peer's outside interface IP address:

```
pixP(config)# crypto map peerQ 10 set peer 192.168.Q.2
```

(where *Q* = peer pod number)

5. Specify the transform set used to reach the peer. Use the transform set name you configured in sub-step 2.

```
pixP(config)# crypto map peerQ 10 set transform-set pixQ
```

(where *Q* = peer pod number)

6. Apply the crypto map set to the outside interface:

```
pixP(config)# crypto map peerQ interface outside
```

(where *Q* = peer pod number)

Task 4: Test and Verify IPSec Configuration

Perform the following steps to test and verify VPN configuration:

Step 1 Verify the IKE policy you just created. Note the default values.

```
pixP(config)# show isakmp
isakmp enable outside
isakmp key cisco123 address 192.168.Q.2 netmask 255.255.255.255
isakmp policy 10 authentication pre-share
isakmp policy 10 encryption des
isakmp policy 10 hash sha
isakmp policy 10 group 1
isakmp policy 10 lifetime 86400
```


What five policy items are configured in an IKE policy?
A: authentication method, encryption algorithm, hash algorithm, D-H group, and ISAKMP SA lifetime.

Which IKE policy value did you configure in a previous step?
A: authentication method as pre-share

Which IKE policy values had defaults?
A: encryption algorithm=des, hash algorithm=sha, D-H group=group 1, and ISAKMP SA lifetime=86400.

Step 2 Examine the IKE policies in your PIX Firewall.

```
pixP(config)# show isakmp policy
Protection suite of priority 10
        encryption algorithm:   DES - Data Encryption Standard (56 bit keys).
        hash algorithm:         Secure Hash Standard
        authentication method:  Pre-Shared Key
        Diffie-Hellman group:   #1 (768 bit)
        lifetime:               86400 seconds, no volume limit
Default protection suite
        encryption algorithm:   DES - Data Encryption Standard (56 bit keys).
        hash algorithm:         Secure Hash Standard
        authentication method:  Rivest-Shamir-Adleman Signature
        Diffie-Hellman group:   #1 (768 bit)
        lifetime:               86400 seconds, no volume limit
```

How did the Default protection suite get configured?
A: It is part of the default configuration.

Step 3 Verify the crypto access list. The list shown is for PIX2 connecting to PIX1:

```
pix2(config)# show access-list
access-list 101 permit ip host 192.168.P.10 host 192.168.Q.10
```

Step 4 Verify correct IPSec parameters (IKE phase two):

```
pixP(config)# show crypto ipsec transform-set
Transform set pixQ: { esp-des  }
   will negotiate = { Tunnel,  },
```

Step 5 Verify correct crypto map configuration. The crypto map shown is for PIX1:

```
pix1(config)# show crypto map
Crypto Map: “peer2” interface: “outside” local address: 192.168.1.2
Crypto Map “peer2” 10 ipsec-isakmp
  Peer = 192.168.2.2
  access-list 101 permit ip host 192.168.1.10 host 192.168.2.10 (hitcnt=0)
  Current peer: 192.168.2.2
  Security association lifetime: 4608000 kilobytes/28800 seconds
  PFS (Y/N): N
  Transform sets={ pix2, }
```

Step 6 Turn on debugging for IPSec and ISAKMP:

```
pixP(config)# debug crypto ipsec
pixP(config)# debug crypto isakmp
```

Step 7 Ensure to clear the IPSec SA, by using the following command:

```
pixP(config)# clear crypto ipsec sa
```

Step 8 Initiate a Web session from your internal Windows NT Server to the internal Windows NT server 192.168.Q.10 of an opposite pod group. Observe the debug output and verify the Web session was established. The debug should state the following to status indicating that IPSec was successful

```
return status is IKMP_NO_ERROR
```

Step 9 Ensure that traffic between peers is being encrypted by performing the following sub-steps:

1. Examine the IPSec security associations. Note the number of packets encrypted and decrypted:

```
pix1(config)# show crypto ipsec sa

interface: outside
    Crypto map tag: peer2, local addr. 192.168.1.2

local  ident (addr/mask/prot/port): (192.168.1.10/255.255.255.255/0/0)
   remote ident (addr/mask/prot/port): (192.168.2.10/255.255.255.255/0/0)
   current_peer: 192.168.2.2
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 210, #pkts encrypt: 210, #pkts digest 0
    #pkts decaps: 201, #pkts decrypt: 227, #pkts verify 0
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0, #pkts decompress failed: 0
    #send errors 29, #recv errors 0
```

2. Generate additional traffic by clicking on the Reload button of your Web browser.

3. Examine the IPSec security associations again. Note that the packet counters have incremented.

```
pix2(config)# show cry ipsec sa

interface: outside
    Crypto map tag: peer2, local addr. 192.168.1.2

local  ident (addr/mask/prot/port): (192.168.2.10/255.255.255.255/0/0)
   remote ident (addr/mask/prot/port): (192.168.3.10/255.255.255.255/0/0)
   current_peer: 192.168.2.2
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 238, #pkts encrypt: 238, #pkts digest 0
    #pkts decaps: 239, #pkts decrypt: 267, #pkts verify 0
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0, #pkts decompress failed: 0
    #send errors 31, #recv errors 0
```

Completion Criteria

You completed this lab exercise if you were able to do the following:

- Cause an IPSec tunnel to be established between PIX Firewalls
- Closely match your PIX Firewall configuration with the example configuration at the end of this laboratory exercise

Example Configurations

The following tables show an example configuration for PIX1 and PIX2. You may experience differences between the example configuration and your own configuration.

PIX1 Example Configuration

The example in the following table is a summary of the configuration for PIX1.

Table 12-1. PIX1 Example Configuration

Example Configuration	Description
ip address outside 192.168.1.2 255.255.255.0 **ip address inside 10.0.1.1 255.255.255.0** **ip address dmz 172.16.1.1 255.255.0.0**	Configures the IP addresses for each PIX Firewall interface.
global (outside) 1 192.168.1.10-192.168.1.254 netmask 255.255.255.0	Creates a global pool on the outside interface.
nat (inside) 1 10.0.0.0 0.0.0.0 0 0	Enables NAT for the inside interface.
static (inside,outside) 192.168.1.10 10.0.1.3 netmask 255.255.255.255 0 0	Creates a static translation between the global IP address o1 192.168.1.10 and the inside Windows NT server at address 10.0.1.3.
access-list 101 permit ip host 192.168.1.10 host 192.168.2.10	The crypto access list specifies that traffic between the internal Windows NT servers of PIX1 and PIX2 be encrypted. The source and destination IP addresses are the global IP addresses of the static translations. Note that the access lists for PIX1 and PIX2 are mirror images of each other.
conduit permit icmp any any **conduit permit tcp host 192.168.1.10 eq www any**	The conduits permit ICMP and Web access for testing.
route outside 0.0.0.0 0.0.0.0 192.168.1.1 1	Specifies the router on the outside interface for the default route.
sysopt connection permit-ipsec	Enables IPSec to bypass access list, access, and conduit restrictions.
crypto ipsec transform-set pix2 esp-des	Defines a crypto map transform set named *pix2* to use **esp-des**.
crypto map peer2 10 ipsec-isakmp	Defines the crypto map named *peer2* with a priority of 10 to use ISAKMP access. The crypto map defines IPSec (IKE phase two) parameters.
crypto map peer2 10 match address 101	Defines the crypto map named *peer2* to use access list 101 for crypto traffic selection.
crypto map peer2 10 set peer 192.168.2.2	Defines the crypto map named *peer2* to point to the peer (pix2) by specifying the peer PIX's outside interface IP address.
crypto map peer2 10 set transform-set pix2	Defines the crypto map named *peer2* to use the transform set named *pix2*.

Example Configuration	Description
crypto map peer2 interface outside	Assigns the crypto map set named *peer2* to the outside PIX interface. As soon as the crypto map is assigned to the interface, the IKE and IPSec policy is active.
isakmp enable outside	Enables ISAKMP (IKE) on the outside interface.
isakmp key cisco123 address 192.168.2.2 netmask 255.255.255.255	Defines the pre-shared IKE key of *cisco123* to work with the IPSec peer at address 192.168.2.2. The address points to the peer's outside interface. A wildcard address of 0.0.0.0 with a netmask of 0.0.0.0 could also have been used.
isakmp policy 10 authentication pre-share	Defines the ISAKMP (IKE) policy of *10* to use pre-shared keys for authentication.
isakmp policy 10 encryption des	Defines the ISAKMP (IKE) policy of *10* to use DES encryption. Could have used 3DES for stronger encryption.
isakmp policy 10 hash sha	Defines the ISAKMP (IKE) policy of *10* to use the SHA-1 hashing algorithm for encryption.
isakmp policy 10 group 1	Specifies use of D-H group 1. Could have used D-H group 2 for stronger security, but requires more CPU time to execute.
isakmp policy 10 lifetime 86400	Specifies an ISAKMP (IKE) lifetime of 86,400 seconds.

PIX2 Example Configuration

The example in the following table is a summary of the configuration for PIX2.

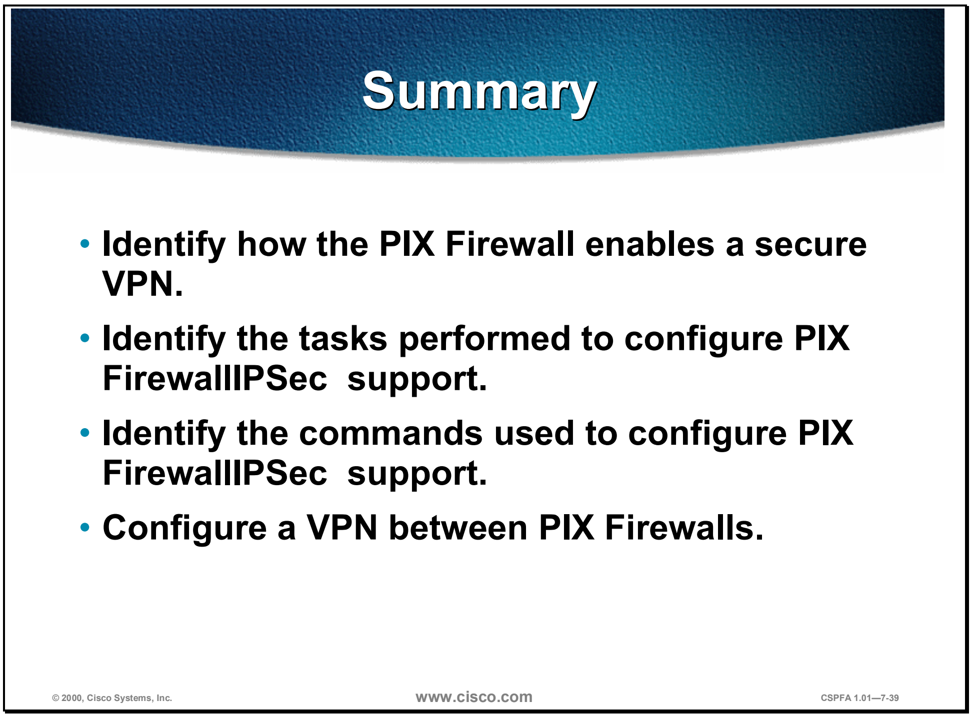
Table 12-2. PIX2 Example Configuration

Example Configuration	Description
ip address outside 192.168.2.2 255.255.255.0 **ip address inside 10.0.2.1 255.255.255.0** **ip address dmz 172.16.2.1 255.255.0.0**	Configures the IP addresses for each PIX Firewall interface.
global (outside) 1 192.168.2.10-192.168.2.254 netmask 255.255.255.0	Creates a global pool on the outside interface.
nat (inside) 1 10.0.0.0 0.0.0.0 0 0	Enables NAT for the inside interface.
static (inside,outside) 192.168.2.10 10.0.2.3 netmask 255.255.255.255 0 0	Creates a static translation between the global IP address o1 192.168.2.10 and the inside Windows NT server at address 10.0.2.3.
access-list 101 permit ip host 192.168.2.10 host 192.168.1.10	The crypto access list specifies that traffic between the internal Windows NT servers of PIX1 and PIX2 be encrypted. The source and destination IP addresses are the global IP addresses of the static translations. Note that the access lists for PIX1 and PIX2 are mirror images of each other.
conduit permit icmp any any **conduit permit tcp host 192.168.2.10 eq www any**	The conduits permit ICMP and Web access for testing.
route outside 0.0.0.0 0.0.0.0 192.168.2.1 1	Specifies the router on the outside interface for the default route.
sysopt connection permit-ipsec	Enables IPSec to bypass access list, access, and conduit restrictions.

Example Configuration	Description
crypto ipsec transform-set pix1 esp-des	Defines a crypto map transform set named *pix1* to use **esp-des**.
crypto map peer1 10 ipsec-isakmp	Defines the crypto map named *peer1* with a priority of 10 to use ISAKMP access. The crypto map defines IPSec (IKE phase two) parameters.
crypto map peer1 10 match address 101	Defines the crypto map named *peer1* to use access list 101 for crypto traffic selection.
crypto map peer1 10 set peer 192.168.1.2	Defines the crypto map named *peer1* to point to the peer (pix1) by specifying the peer PIX's outside interface IP address.
crypto map peer1 10 set transform-set pix1	Defines the crypto map named *peer1* to use the transform set named *pix1*.
crypto map peer1 interface outside	Assigns the crypto map set named *peer1* to the outside PIX interface. As soon as the crypto map is assigned to the interface, the IKE and IPSec policy is active.
isakmp enable outside	Enables ISAKMP (IKE) on the outside interface.
isakmp key cisco123 address 192.168.1.2 netmask 255.255.255.255	Defines the pre-shared IKE key of *cisco123* to work with the IPSec peer at address 192.168.1.2. The address points to the peer's outside interface. A wildcard address of 0.0.0.0 with a netmask of 0.0.0.0 could also have been used.
isakmp policy 10 authentication pre-share	Defines the ISAKMP (IKE) policy of *10* to use pre-shared keys for authentication.
isakmp policy 10 encryption des	Defines the ISAKMP (IKE) policy of *10* to use DES encryption. Could have used 3DES for stronger encryption.
isakmp policy 10 hash sha	Defines the ISAKMP (IKE) policy of *10* to use the SHA-1 hashing algorithm for encryption.
isakmp policy 10 group 1	Specifies use of D-H group 1. Could have used D-H group 2 for stronger security, but requires more CPU time to execute.
isakmp policy 10 lifetime 86400	Specifies an ISAKMP (IKE) lifetime of 86,400 seconds.

Summary

This section summarizes the tasks you learned to complete in this chapter.



The Cisco IOS Firewall Context-Based Access Control Configuration

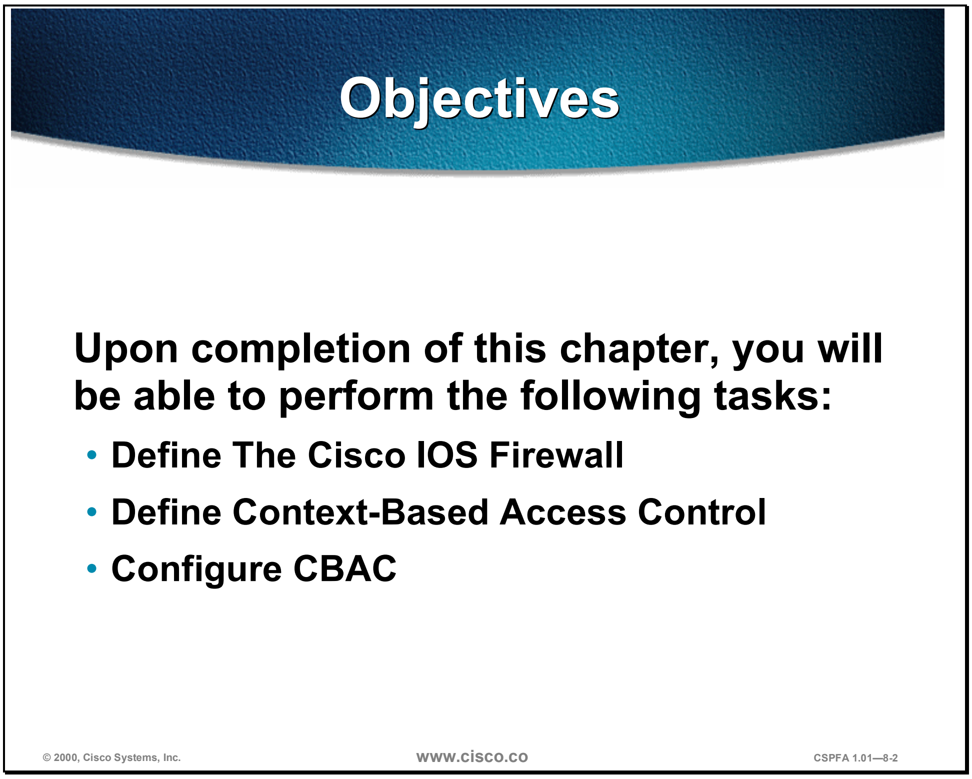
Overview

This chapter includes the following topics:

- Objectives
- Introduction to The Cisco IOS Firewall
- Context-Based Access Control
- Basic router security
- Audit trail and alert
- Global timeouts and thresholds
- Port-to-Application Mapping (PAM)
- Define inspection rules
- Apply inspection rules and ACLs to router interfaces
- Test and verify
- Lab exercise
- Summary

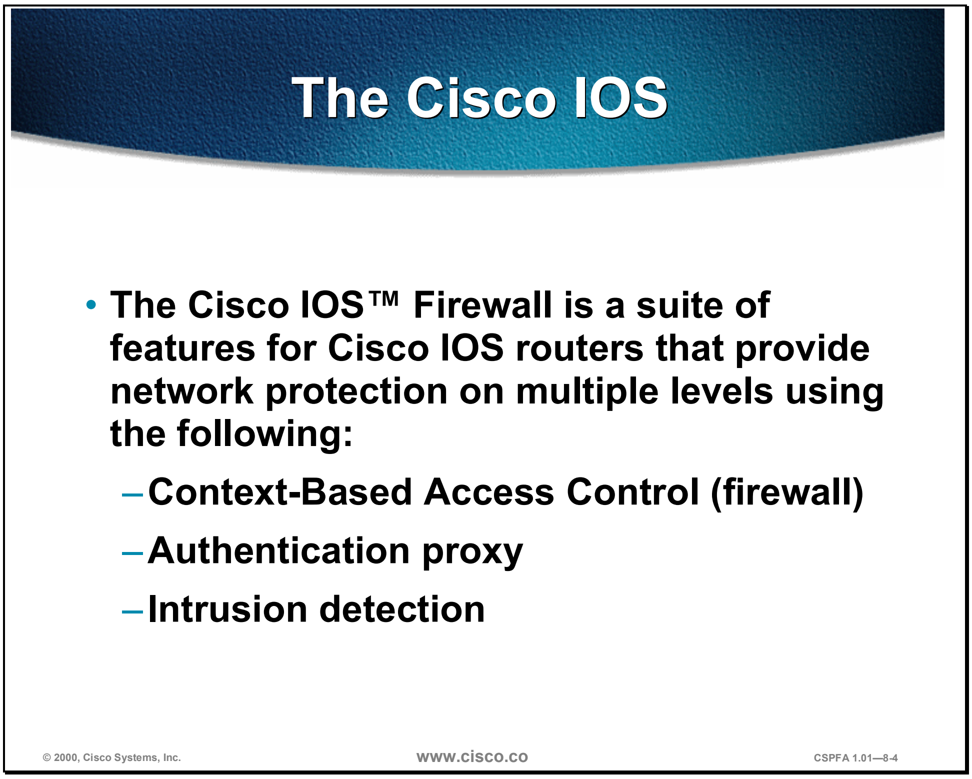
Objectives

This section lists the chapter's objectives.



Introduction to The Cisco IOS Firewall

This section introduces the features of The Cisco IOS™ (IOS) Firewall.



The Cisco IOS Firewall is a security-specific option for Cisco IOS software. It integrates robust firewall functionality, authentication proxy, and intrusion detection for every network perimeter, and enriches existing Cisco IOS security capabilities. It adds greater depth and flexibility to existing Cisco IOS security solutions, such as authentication, encryption, and failover, by delivering state-of-the-art security features such as stateful, application-based filtering; dynamic per-user authentication and authorization; defense against network attacks; Java blocking; and real-time alerts. When combined with Cisco IOS IPSec software and other Cisco IOS software-based technologies, such as Layer 2 Tunneling Protocol (L2TP) tunneling and quality of service (QoS), the Cisco IOS Firewall provides a complete, integrated virtual private network (VPN) solution.

Context-Based Access Control

The Cisco IOS Firewall Context-Based Access Control (CBAC) engine provides secure, per-application access control across network perimeters. CBAC enhances security for TCP and UDP applications that use well-known ports, such as FTP and e-mail traffic, by scrutinizing source and destination addresses. CBAC allows network administrators to implement firewall intelligence as part of an integrated, single-box solution.

For example, sessions with an extranet partner involving Internet applications, multimedia applications, or Oracle databases would no longer need to open a network doorway accessible via weaknesses in a partner's network. CBAC lets tightly secured networks run today's basic application traffic, as well as advanced applications such as multimedia and videoconferencing, securely through a router.

Authentication Proxy

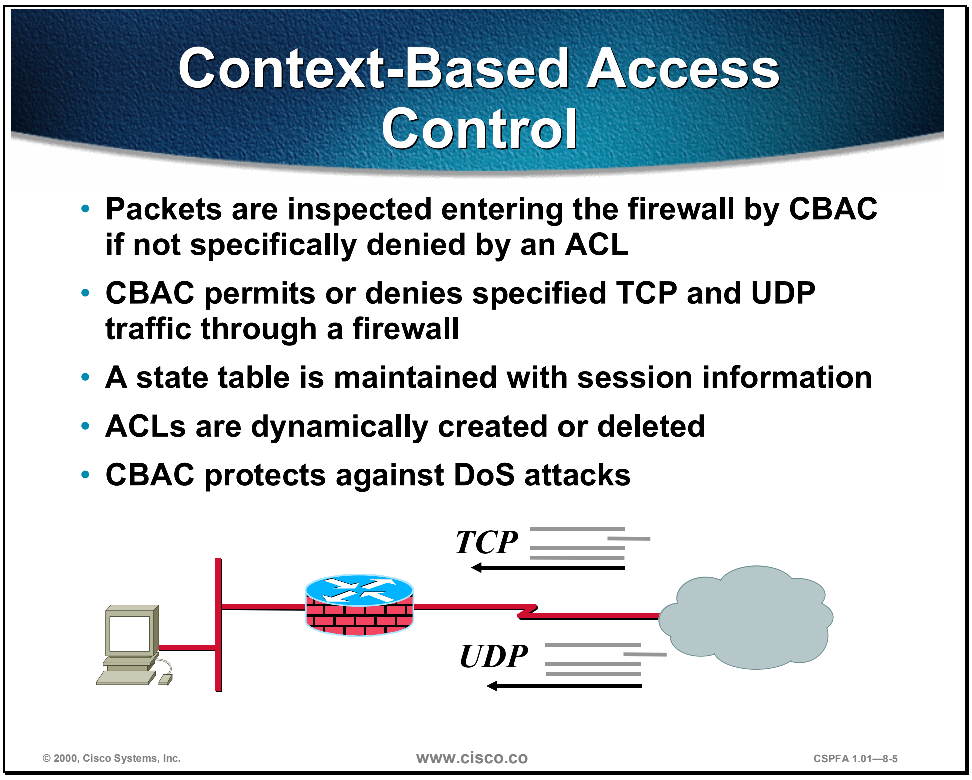
Network administrators can create specific security policies for each user with Cisco IOS Firewall LAN-based, dynamic, per-user authentication and authorization. Previously, user identity and related authorized access were determined by a user's fixed IP address, or a single security policy had to be applied to an entire user group or subnet. Now, per-user policy can be downloaded dynamically to the router from a TACACS+ or RADIUS authentication server using Cisco IOS software authentication, authorization, and accounting (AAA) services.

Users can log into the network or access the Internet via HTTP, and their specific access profiles will automatically be downloaded. Appropriate dynamic individual access privileges are available as required, protecting the network against more general policies applied across multiple users. Authentication and authorization can be applied to the router interface in either direction to secure inbound or outbound extranet, intranet, and Internet usage.

Intrusion Detection

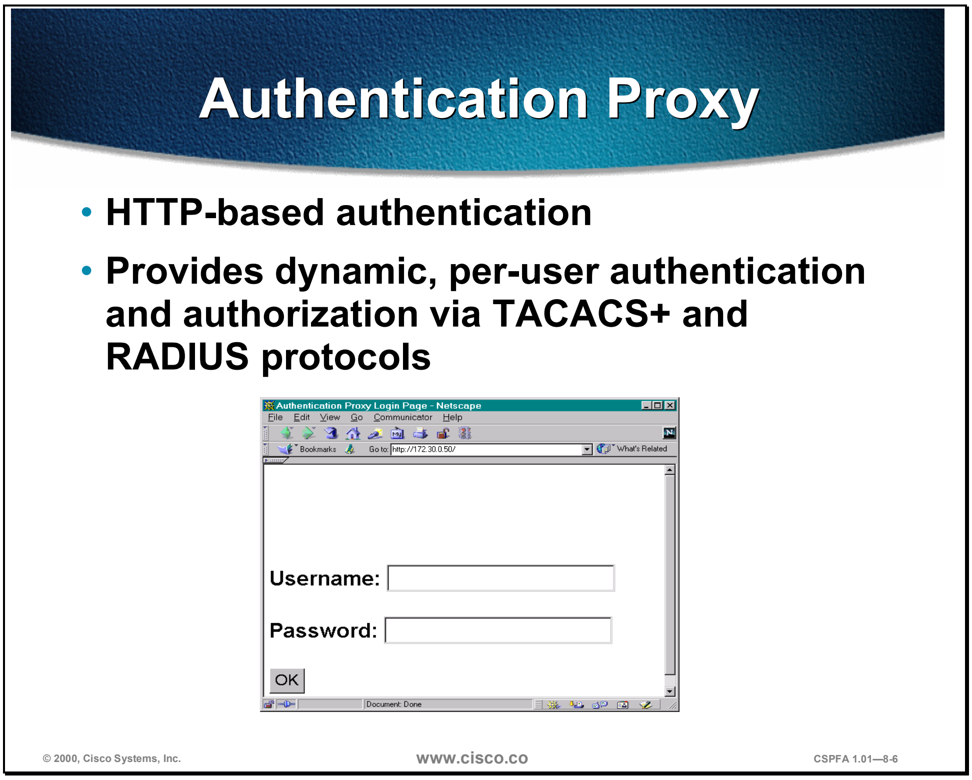
Intrusion detection systems (IDS) provide a level of protection beyond the firewall by protecting the network from internal and external attacks and threats. Cisco IOS Firewall IDS technology enhances perimeter firewall protection by taking appropriate action on packets and flows that violate the security policy or represent malicious network activity.

Cisco IOS Firewall intrusion detection capabilities are ideal for providing additional visibility at intranet, extranet, and branch-office Internet perimeters. Network administrators now enjoy more robust protection against attacks on the network, and can automatically respond to threats from internal or external hosts.



CBAC intelligently filters TCP and UDP packets based on application-layer protocol session information. It can inspect traffic for sessions that originate on any interface of the router. CBAC inspects traffic that travels through the firewall to discover and manage state information for TCP and UDP sessions. This state information is used to create temporary openings in the firewall's access lists to allow return traffic and additional data connections for permissible sessions.

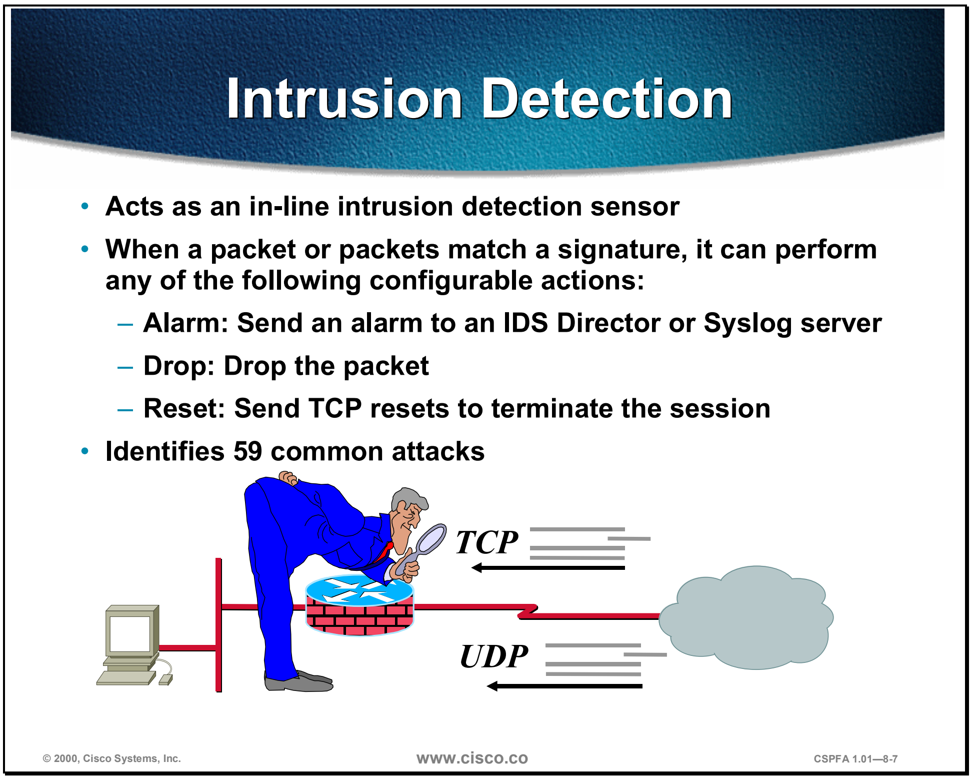
Inspecting packets at the application layer, and maintaining TCP and UDP session information, provides CBAC with the ability to detect and prevent certain types of network attacks such as SYN flooding. CBAC also inspects packet sequence numbers in TCP connections to see if they are within expected ranges—CBAC drops any suspicious packets. Additionally, CBAC can detect unusually high rates of new connections and issue alert messages. CBAC inspection can help protect against certain Denial of Service (DoS) attacks involving fragmented IP packets. Even though the firewall prevents an attacker from making actual connections to a given host, the attacker can disrupt services provided by that host. This is done by sending many non-initial IP fragments or by sending complete fragmented packets through a router with an access control list (ACL) that filters the first fragment of a fragmented packet. These fragments can tie up resources on the target host as it tries to reassemble the incomplete packets.



The Cisco IOS Firewall authentication proxy feature allows network administrators to apply specific security policies on a per-user basis. Previously, user identity and related authorized access was associated with a user's IP address, or a single security policy had to be applied to an entire user group or subnet. Now, users can be identified and authorized on the basis of their per-user policy, and access privileges tailored on an individual basis are possible, as opposed to general policy applied across multiple users.

With the authentication proxy feature, users can log in to the network or access the Internet via HTTP, and their specific access profiles are automatically retrieved and applied from a Cisco Secure Asynchronous Communications Server (ACS), or other RADIUS or TACACS+ authentication server. The user profiles are active only when there is active traffic from the authenticated users.

The authentication proxy is compatible with other Cisco IOS security features such as Network Address Translation (NATCBAC, IP Security (IPSec) encryption, and VPN client software.



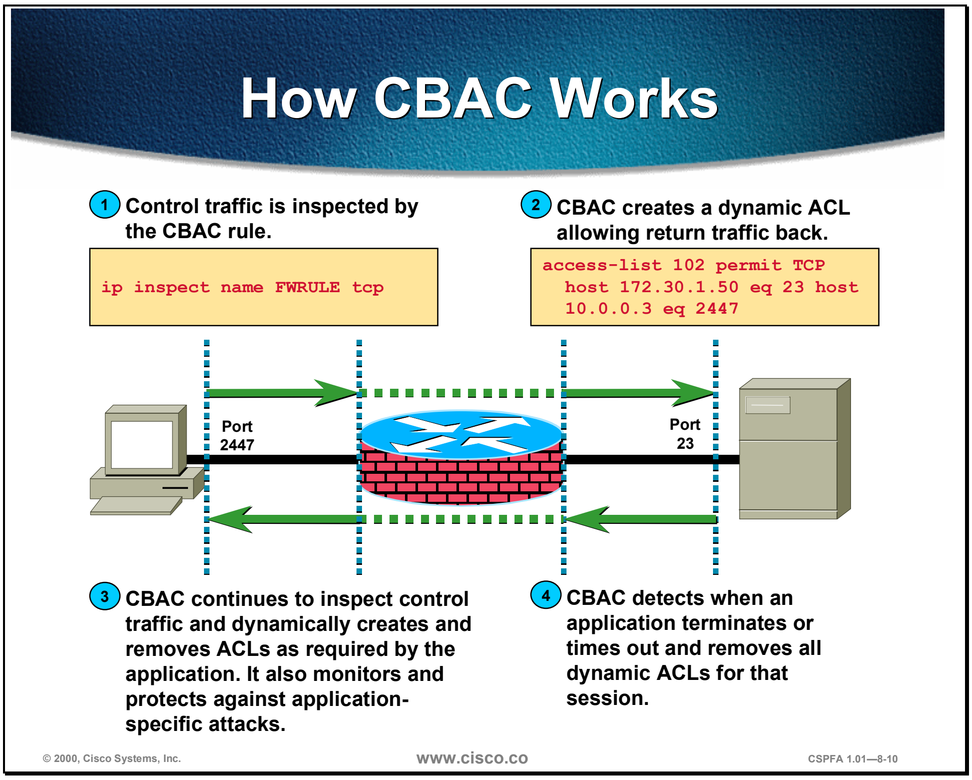
The Cisco IOS Firewall now offers intrusion detection technology for mid-range and high-end router platforms with firewall support. It is ideal for any network perimeter, and especially for locations in which a router is being deployed and additional security between network segments is required. It also can protect intranet and extranet connections where additional security is mandated, and branch-office sites connecting to the corporate office or Internet.

The Cisco IOS Firewall's intrusion detection system identifies 59 common attacks using signatures to detect patterns of misuse in network traffic. The intrusion detection signatures available in the new release of the Cisco IOS Firewall were chosen from a broad cross-section of intrusion detection signatures. The signatures represent severe breaches of security and the most common network attacks and information-gathering scans.

Context-Based Access Controls

This section describes the limitations of Cisco IOS access control lists and explains how CBAC better protects users from attack. It also lists the protocols supported by CBAC and describes the added alert and audit trail features. Finally, the CBAC configuration tasks are listed.

Before delving into CBAC, some basic ACL concepts need to be covered briefly. An ACL provides packet filtering: it has an implied deny all at the end of the ACL and if the ACL is not configured, it permits all connections. Without CBAC, traffic filtering is limited to access list implementations that examine packets at the network layer, or at most, the transport layer.



With CBAC, you specify which protocols you want to be inspected, and you specify an interface and interface direction (in or out) where inspection originates. Only specified protocols will be inspected by CBAC. For these protocols, packets flowing through the firewall in any direction are inspected, as long as they flow through the interface where inspection is configured. Packets entering the firewall are inspected by CBAC only if they first pass the inbound access list at the interface. If a packet is denied by the ACL, the packet is simply dropped and not inspected by CBAC.

CBAC inspects and monitors only the control channels of connections; the data channels are not inspected. For example, during FTP sessions both the control and data channels (which are created when a data file is transferred) are monitored for state changes, but only the control channel is inspected (that is, the CBAC software parses the FTP commands and responses).

CBAC inspection recognizes application-specific commands in the control channel, and detects and prevents certain application-level attacks. CBAC inspection tracks sequence numbers in all TCP packets, and drops those packets with sequence numbers that are not within expected ranges. CBAC inspection recognizes application-specific commands (such as illegal Simple Mail Transfer Protocol [SMTP] commands) in the control channel, and detects and prevents certain application-level attacks. When CBAC suspects an attack, the DoS feature can take several actions:

- Generate alert messages
- Protect system resources that could impede performance
- Block packets from suspected attackers

CBAC uses timeout and threshold values to manage session state information, helping to determine when to drop sessions that do not become fully established.

Setting timeout values for network sessions helps prevent DoS attacks by freeing up system resources, dropping sessions after a specified amount of time. Setting threshold values for network sessions helps prevent DoS attacks by controlling the number of half-open sessions, which limits the amount of system resources applied to half-open sessions. When a session is dropped, CBAC sends a reset message to the devices at both endpoints (source and destination) of the session. When the system under DoS attack receives a reset command, it releases, or frees up, processes and resources related to that incomplete session.

CBAC provides three thresholds against DoS attacks:

- The total number of half-open TCP or UDP sessions
- The number of half-open sessions based on time
- The number of half-open TCP-only sessions per host

If a threshold is exceeded, CBAC has two options:

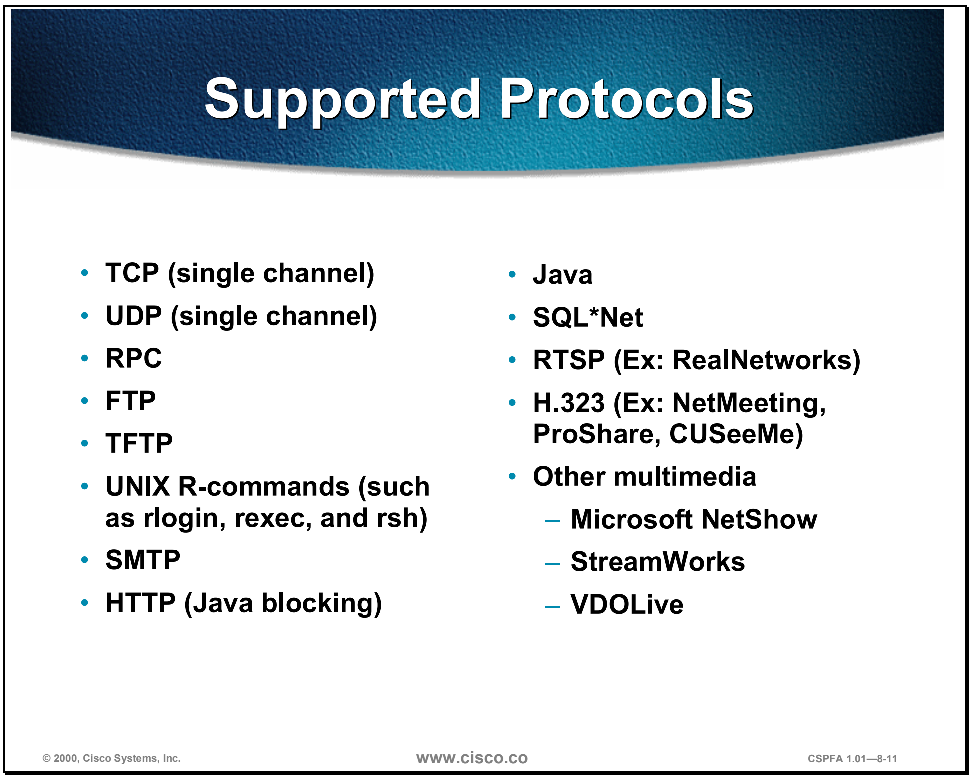
- Send a reset message to the endpoints of the oldest half-open session, making resources available to service newly arriving SYN packets.
- In the case of half-open TCP-only sessions, CBAC blocks all SYN packets temporarily for the duration configured by the threshold value. When the router blocks a SYN packet, the TCP three-way handshake is never initiated, which prevents the router from using memory and processing resources needed for valid connections.

DoS detection and prevention requires that you create a CBAC inspection rule and apply that rule on an interface. The inspection rule must include the protocols that you want to monitor against DoS attacks. For example, if you have TCP inspection enabled on the inspection rule, then CBAC can track all TCP connections to watch for DoS attacks. If the inspection rule includes FTP protocol inspection but not TCP inspection, CBAC tracks only FTP connections for DoS attacks.

A state table maintains session state information. Whenever a packet is inspected, a state table is updated to include information about the state of the packet's connection. Return traffic will only be permitted back through the firewall if the state table contains information indicating that the packet belongs to a permissible session. Inspection controls the traffic that belongs to a valid session and forwards the traffic it does not know. When return traffic is inspected, the state table information is updated as necessary.

UDP sessions are approximated. With UDP there are no actual sessions, so the software approximates sessions by examining the information in the packet and determining if the packet is similar to other UDP packets (for example, similar source or destination addresses and port numbers), and if the packet was detected soon after another, similar UDP packet. Soon means within the configurable UDP idle timeout period.

Access list entries are dynamically created and deleted. CBAC dynamically creates and deletes access list entries at the firewall interfaces, according to the information maintained in the state tables. These access list entries are applied to the interfaces to examine traffic flowing back into the internal network. These entries create temporary openings in the firewall to permit only traffic that is part of a permissible session. The temporary access list entries are never saved to nonvolatile RAM (NVRAM.)



You can configure CBAC to inspect the following types of sessions:

- All TCP sessions, regardless of the application-layer protocol (sometimes called single-channel or generic TCP inspection)
- All UDP sessions, regardless of the application-layer protocol (sometimes called single-channel or generic UDP inspection)

You can also configure CBAC to specifically inspect certain application-layer protocols. The following application-layer protocols can all be configured for CBAC:

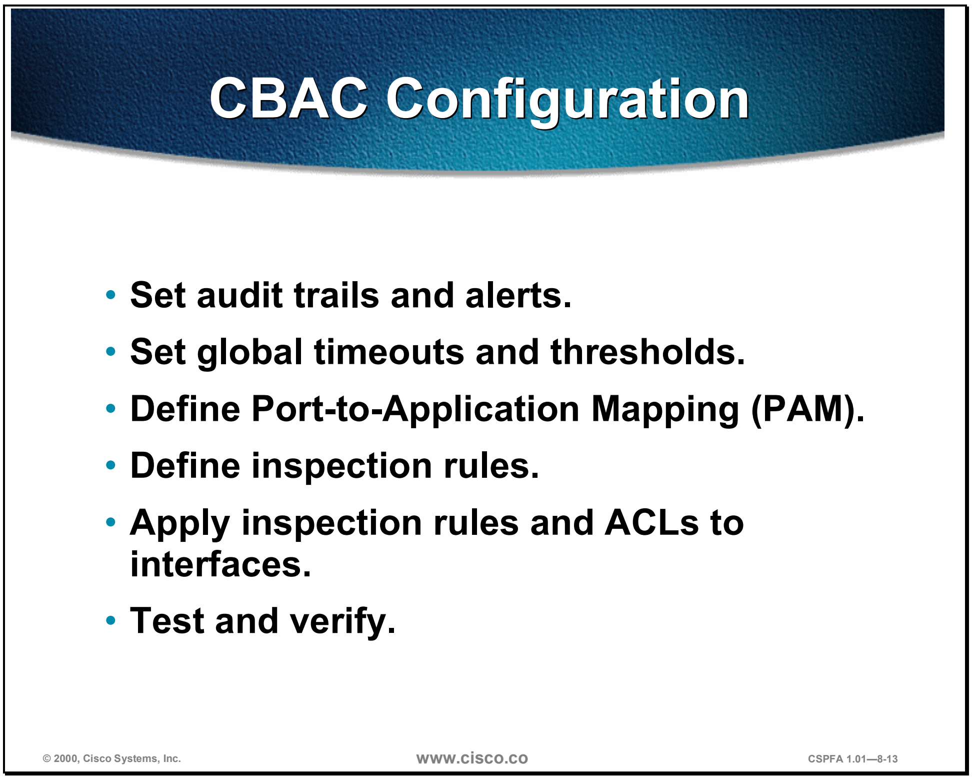
- RPC (Sun RPC, not DCE RPC)
- Microsoft RPC
- FTP
- TFTP
- UNIX R-commands (such as rlogin, rexec, and rsh)
- SMTP
- HTTP (Java blocking)
- Java
- SQL*Net
- RTSP (Ex: RealNetworks)
- H.323 (Ex: NetMeeting, ProShare, CU-SeeMe [only the White Pine version])
- Microsoft NetShow
- StreamWorks

- VDOLive

When a protocol is configured for CBAC, that protocol traffic is inspected, state information is maintained, and in general, packets are allowed back through the firewall only if they belong to a permissible session.

CBAC also generates real-time alerts and audit trails based on events tracked by the firewall. Enhanced audit trail features use Syslog to track all network transactions; recording time stamps, source host, destination host, ports used, and the total number of transmitted bytes, for advanced, session-based reporting.

Real-time alerts send Syslog error messages to central management consoles upon detecting suspicious activity. Using CBAC inspection rules, you can configure alerts and audit trail information on a per-application protocol basis. For example, if you want to generate audit trail information for HTTP traffic, you can specify that in the CBAC rule covering HTTP inspection.

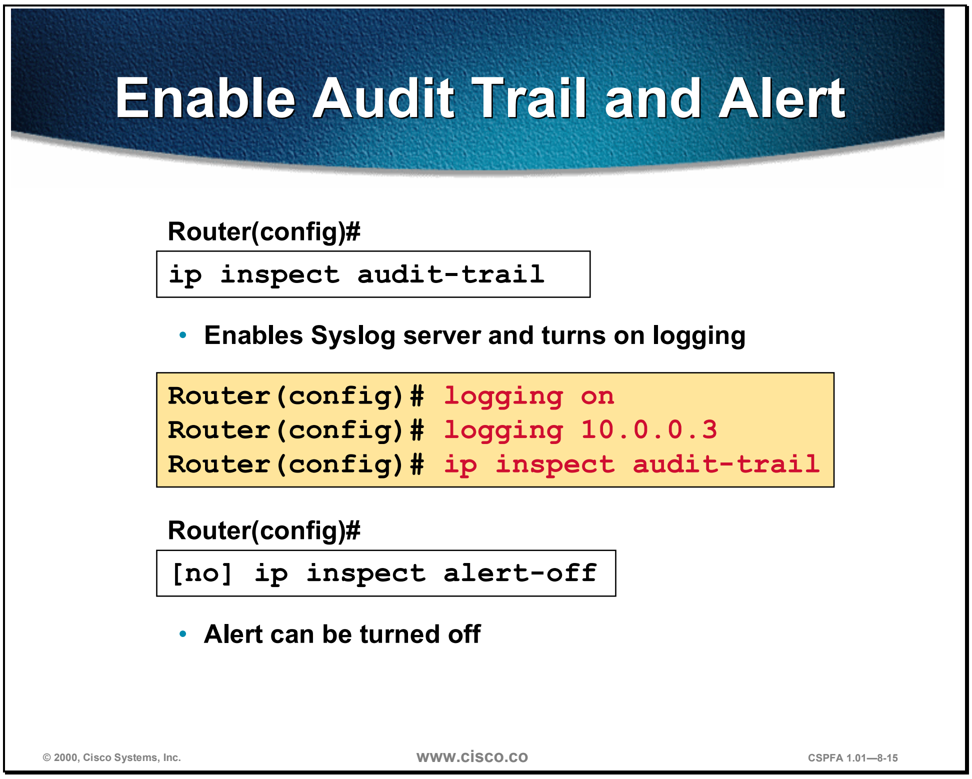


The following are the tasks used to configure CBAC:

- Set audit trails and alerts.
- Set global timeouts and thresholds.
- Define Port-to-Application Mapping (PAM).
- Define inspection rules.
- Apply inspection rules and ACLs to interfaces.
- Test and verify.

Audit Trail and Alert

This section discusses how to configure an audit trail and alert.



Turn on logging and audit trail to provide a record of network access through the firewall, including illegitimate access attempts, and inbound and outbound services.

Use the **ip inspect audit-trail** and **ip inspect alert-off** commands to enable audit trail and alert, respectively.

The syntax for the **ip inspect audit-trail** commands is as follows:

ip inspect audit-trail

no ip inspect audit trail

The syntax for the **ip inspect alert-off** commands is as follows:

ip inspect alert-off

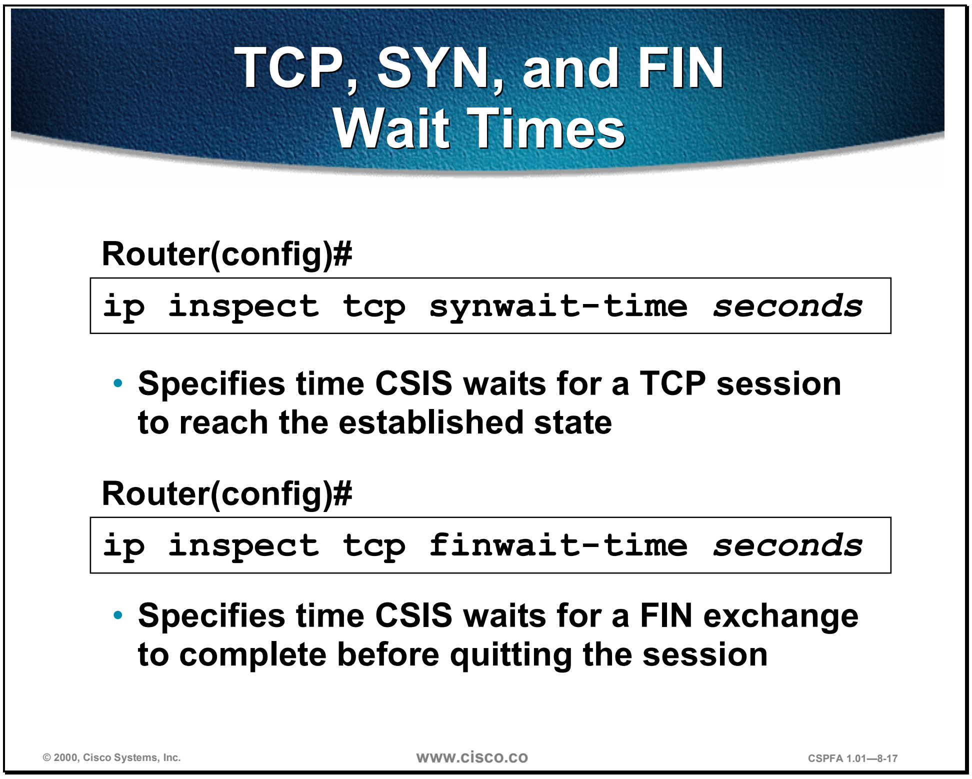
no ip inspect alert-off

No other arguments or keywords are used with either command.

Global Timeouts and Thresholds

This section discusses how to configure the following global timeouts and thresholds:

- TCP, SYN, and FIN wait times
- TCP, UDP, and Domain Name System (DNS) idle times
- TCP flood DoS protection



CBAC uses timeouts and thresholds to determine how long to manage state information for a session, and to determine when to drop sessions that do not become fully established. These timeouts and thresholds apply globally to all sessions.

You can use the default timeout and threshold values, or you can change to values more suitable to your security requirements. You should make any changes to the timeout and threshold values before you continue configuring CBAC.

To define how long the software will wait for a TCP session to reach the established state before dropping the session, use the **ip inspect tcp synwait-time** global configuration command. Use the **no** form of this command to reset the timeout to the default.

The syntax of the **ip inspect tcp synwait-time** command is as follows:

ip inspect tcp synwait-time ***seconds***

no ip inspect tcp synwait-time

Arguments	Description
seconds	Specifies how long the software will wait for a TCP session to reach the established state before dropping the session (The default is 30 seconds).

To define how long a TCP session will still be managed after the firewall detects a FIN exchange, use the **ip inspect tcp finwait-time** global configuration command. Use the **no** form of this command to reset the timeout to default.

The syntax of the **ip inspect tcp finwait-time** command is as follows:

ip inspect tcp finwait-time *seconds*

no ip inspect tcp finwait-time

Arguments	Description
seconds	Specifies how long a TCP session will be managed after the firewall detects a FIN exchange (The default is 5 seconds).

TCP, UDP, and DNS Idle Times

Router(config)#

```
ip inspect tcp idle-time seconds
ip inspect udp idle-time seconds
```

- **Specifies time allowed for a TCP or UDP session with no activity**

Router(config)#

```
ip inspect dns-timeout seconds
```

- **Specifies time allowed for a DNS session with no activity**

 www.cisco.co CSPFA 1.01—8-18

To specify the TCP idle timeout (the length of time a TCP session will still be managed after no activity), use the **ip inspect tcp idle-time** global configuration command. Use the **no** form of this command to reset the timeout to default.

To specify the UDP idle timeout (the length of time a UDP session will still be managed after no activity), use the **ip inspect udp idle-time** global configuration command. Use the **no** form of this command to reset the timeout to default.

The syntax for the **ip inspect {tcp | udp} idle-time** commands is as follows:

ip inspect {tcp | udp} idle-time ***seconds***

no ip inspect {tcp | udp} idle-time

Arguments	Description
seconds	Specifies the length of time a TDP or a UCP session will still be managed after no activity. For TCP sessions, the default is 3600 seconds (1 hour). For UDP sessions, the default is 30 seconds.

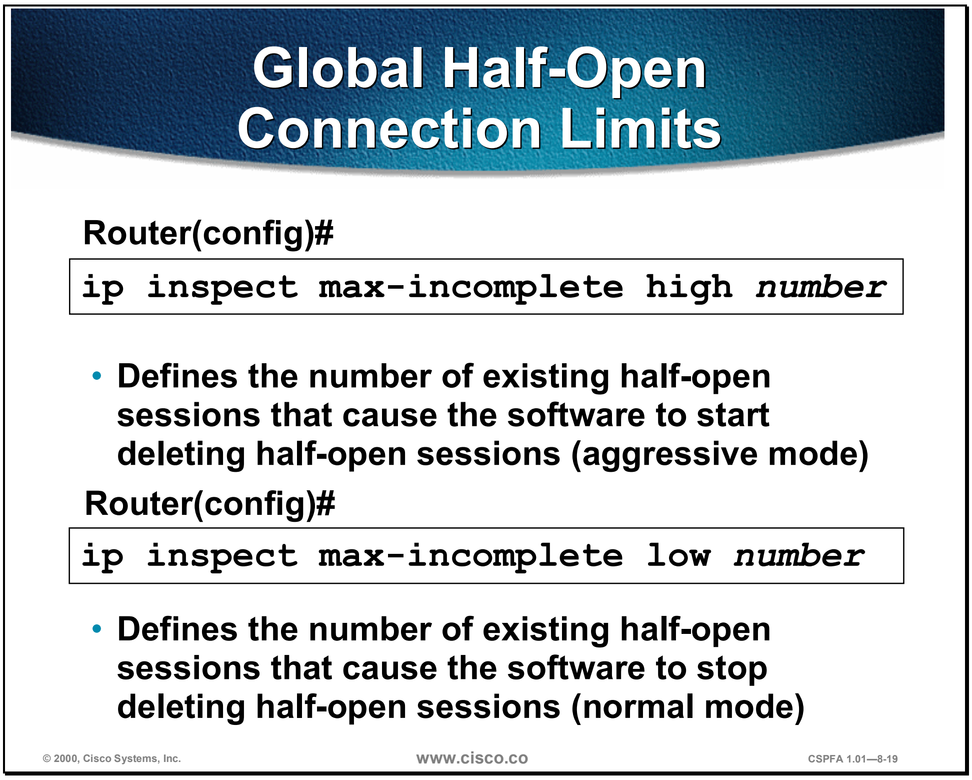
To specify the DNS idle timeout (the length of time a DNS name lookup session will still be managed after no activity), use the **ip inspect dns-timeout** global configuration command. Use the **no** form of this command to reset the timeout to default.

The syntax for the **ip inspect dns-timeout** command is as follows:

ip inspect dns-timeout ***seconds***

no ip inspect dns-timeout

Arguments	Description
seconds	Specifies the length of time a DNS name lookup session will still be managed after no activity (The default is 5 seconds).



An unusually high number of half-open sessions (either absolute or measured as the arrival rate) could indicate that a DoS attack is occurring. For TCP, half-open means that the session has not reached the established state—the TCP three-way handshake has not yet been completed. For UDP, half-open means that the firewall has detected no return traffic.

CBAC measures both the total number of existing half-open sessions and the rate of session establishment attempts. Both TCP and UDP half-open sessions are counted in the total number and rate measurements. Measurements are made once a minute.

When the number of existing half-open sessions rises above a threshold (the **max-incomplete high *number***), CBAC will go in to “aggressive mode” and delete half-open sessions as required to accommodate new connection requests. The software continues to delete half-open requests as necessary, until the number of existing half-open sessions drops below another threshold (the **max-incomplete low *number***).

To define the number of existing half-open sessions that will cause the software to start deleting half-open sessions, use the **ip inspect max-incomplete high** command in global configuration mode. Use the **no** form of this command to reset the threshold to default.

The syntax for the **ip inspect max-incomplete high** command is as follows:

ip inspect max-incomplete high *number*

no ip inspect max-incomplete high *number*

Arguments	Description
high *number*	Specifies the number of existing half-open sessions that will cause the software to start deleting half-open sessions (The default is 500 half-open sessions).

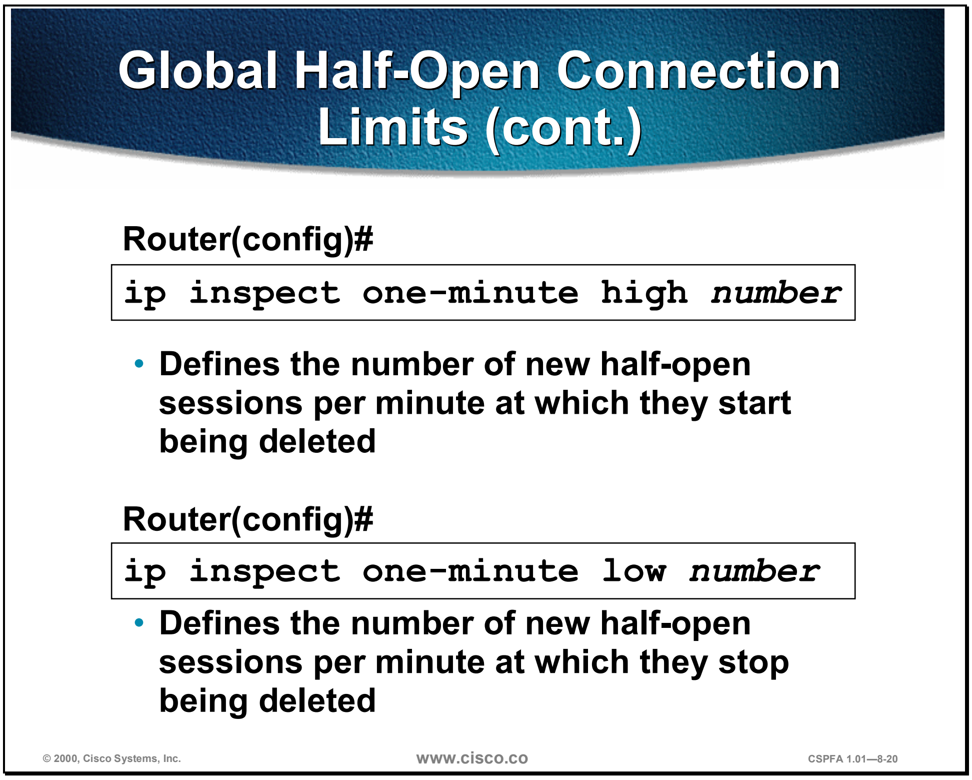
To define the number of existing half-open sessions that will cause the software to stop deleting half-open sessions, use the **ip inspect max-incomplete low** command in global configuration mode. Use the **no** form of this command to reset the threshold to default.

The syntax for the **ip inspect max-incomplete low** command is as follows:

ip inspect max-incomplete low *number*

no ip inspect max-incomplete low *number*

Arguments	Description
low *number*	Specifies the number of existing half-open sessions that will cause the software to stop deleting half-open sessions (The default is 400 half-open sessions).



When the rate of new connection attempts rises above a threshold (the **one-minute high *number***), the software will delete half-open sessions as required to accommodate new connection attempts. The software continues to delete half-open sessions as necessary, until the rate of new connection attempts drops below another threshold (the **one-minute low *number***). The rate thresholds are measured as the number of new session connection attempts detected in the last one-minute sample period. The firewall router reviews the one-minute rate on an ongoing basis, meaning that the router reviews the rate more frequently than one minute and does not keep deleting half-open sessions for one-minute after a DoS attack has stopped—it will be less time.

To define the rate of new unestablished sessions that will cause the software to start deleting half-open sessions, use the **ip inspect one-minute high** command in global configuration mode. Use the **no** form of this command to reset the threshold to default.

The syntax for the **ip inspect one-minute high** command is as follows:

ip inspect one-minute high *number*

no ip inspect one-minute high

Arguments	Description
high *number*	Specifies the rate of new unestablished TCP sessions that will cause the software to start deleting half-open sessions (The default is 500 half-open sessions).

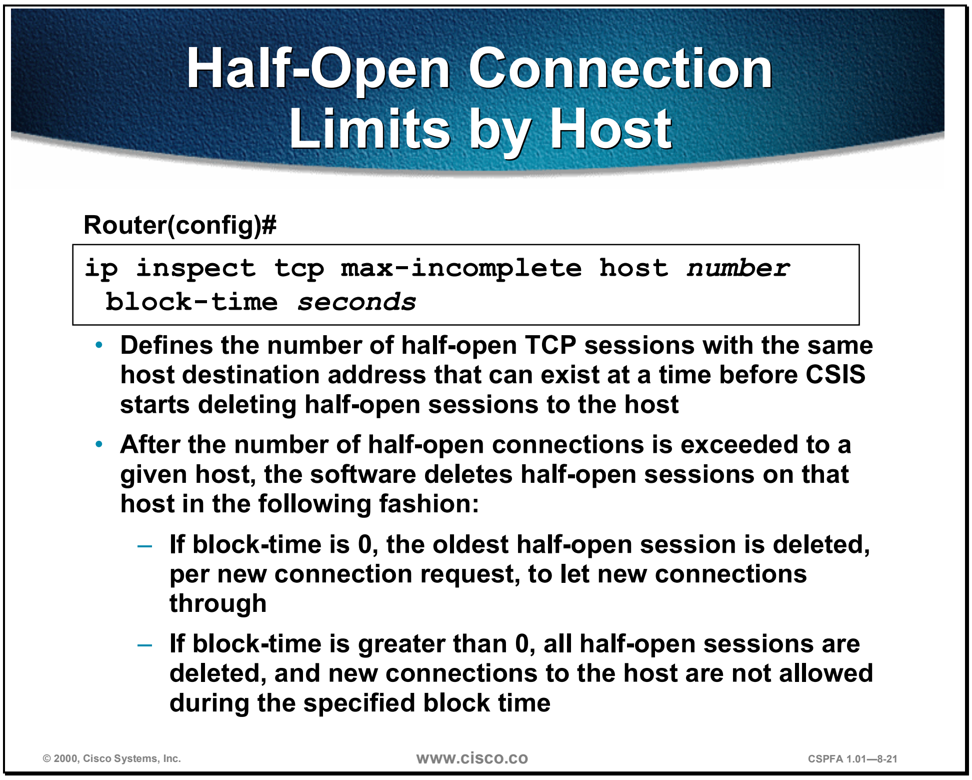
To define the rate of new unestablished TCP sessions that will cause the software to stop deleting half-open sessions, use the **ip inspect one-minute low** command in global configuration mode. Use the **no** form of this command to reset the threshold to the default.

The syntax for the **ip inspect one-minute low** command is as follows:

ip inspect one-minute low*number*

no ip inspect one-minute low

Arguments	Description
low *number*	Specifies the number of existing half-open sessions that will cause the software to stop deleting half-open sessions (The default is 400 half-open sessions).



An unusually high number of half-open sessions with the same destination host address could indicate that a DoS attack is being launched against the host. Whenever the number of half-open sessions with the same destination host address rises above a threshold (the **max-incomplete host *number***), the software will delete half-open sessions according to one of the following methods:

- If the **block-time *seconds*** timeout is 0 (the default)—The software deletes the oldest existing half-open session for the host for every new connection request to the host. This ensures that the number of half-open sessions to a given host will never exceed the threshold.
- If the **block-time *seconds*** timeout is greater than 0—The software deletes all existing half-open sessions for the host, and then blocks all new connection requests to the host. The software will continue to block all new connection requests until the block time expires.

The software also sends Syslog messages whenever the **max-incomplete host *number*** is exceeded, and when blocking of connection initiations to a host starts or ends.

The global values specified for the threshold and blocking time apply to all TCP connections inspected by CBAC.

Use the **ip inspect tcp max-incomplete host** global configuration command to specify threshold and blocking time values for TCP host-specific DoS detection and prevention. Use the **no** form of this command to reset the threshold and blocking time to the default values.

The syntax for the **ip inspect tcp max-incomplete host** command is as follows:

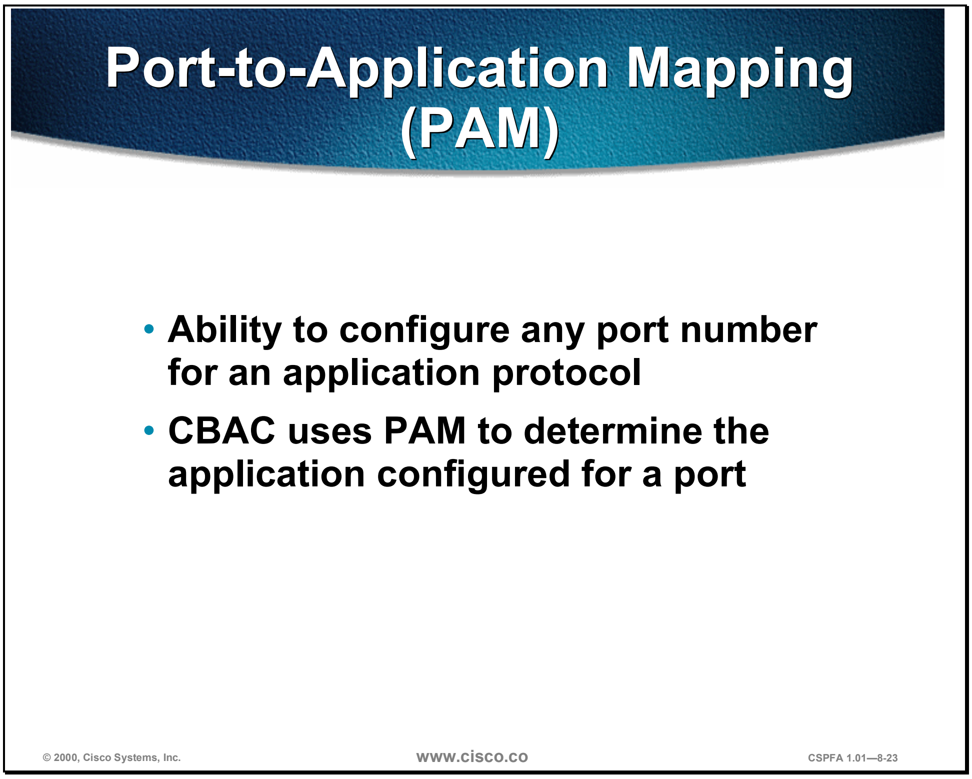
ip inspect tcp max-incomplete host *number* block-time *seconds*

no ip inspect tcp max-incomplete host

Arguments	Description
host ***number***	Specifies how many half-open TCP sessions with the same host destination address can exist at a time before the software starts deleting half-open sessions to the host. Use a number from 1 to 250 (The default is 50 half-open sessions).
block-time ***seconds***	Specifies how long the software will continue to delete new connection requests to the host (The default is 0 secconds).

Port-to-Application Mapping

This section discusses the configuration of port numbers for application protocols.



Port-to-Application Mapping (PAM) allows you to customize TCP or UDP port numbers for network services or applications. PAM uses this information to support network environments that run services using ports that are different from the registered or well-known ports associated with an application.

Using the port information, PAM establishes a table of default port-to-application mapping information at the firewall. The information in the PAM table enables CBAC supported services to run on nonstandard ports. Previously, CBAC was limited to inspecting traffic using only the well-known or registered ports associated with an application. Now, PAM allows network administrators to customize network access control for specific applications and services.

PAM also supports host or subnet-specific port mapping, which allows you to apply PAM to a single host or subnet using standard ACLs. Host- or subnet-specific port mapping is done using standard ACLs.

System-Defined Port Mapping

PAM creates a table, or database, of system-defined mapping entries using the well-known or registered port mapping information set up during the system startup. The system-defined entries comprise all the services supported by CBAC, which requires the system-defined mapping information to function properly.

Note The system-defined mapping information cannot be deleted or changed; that is, you cannot map HTTP services to port 21 (FTP) or FTP services to port 80 (HTTP).

The following lists the default system-defined services and applications in the PAM table.

Application	Port
cuseeme	7648
exec	512
ftp	21
http	80
h323	1720
login	513
mgcp	2427
msrpc	135
netshow	1755
realmedia	7070
rtsp	554
rtsp	8554
shell	514
sip	5060
smtp	25
sql-net	1521
streamworks	1558
sunrpc	111
telnet	23
tftp	69
vdolive	7000

User-Defined Port Mapping

Router(config)#

```
ip port-map appl_name port port_num
```

- **Maps a port number to an application**

Router(config)#

```
access-list permit acl_num ip_addr
ip port-map appl_name port port_num list acl_num
```

- **Maps a port number to an application for a given host**

Router(config)#

```
access-list permit acl_num ip_addr wildcard_mask
ip port-map appl_name port port_num list acl_num
```

- **Maps a port number to an application for a given network**

 www.cisco.co CSPFA 1.01—8-24

Network services or applications that use nonstandard ports require user-defined entries in the PAM table. For example, your network might run HTTP services on the nonstandard port 8000 instead of on the system-defined default port 80. In this case, you can use PAM to map port 8000 with HTTP services. If HTTP services run on other ports, use PAM to create additional port mapping entries. After you define a port mapping, you can overwrite that entry at a later time by simply mapping that specific port with a different application.

Note If you try to map an application to a system-defined port, a message appears warning you of a mapping conflict.

User-defined port mapping information can also specify a range of ports for an application by establishing a separate entry in the PAM table for each port number in the range.

User-defined entries are saved with the default mapping information when you save the router configuration.

To establish PAM, use the **ip port-map** configuration command. Use the **no** form of this command to delete user-defined PAM entries.

The syntax for the **ip port-map** command is as follows:

ip port-map ***appl_name*** **port** ***port_num*** **[list** ***acl_num*****]**

Arguments	Description
appl_name	Specifies the name of the application with which to apply the port mapping. Use one of the following application names: cuseeme, dns, exec, finger, ftp, gopher, http, h323, imap, kerberos, ldap, login, lotusnote, mgcp, msrpc, ms-sql, netshow, nfs, nntp, pop2, pop3, realmedia, rtsp, sap, shell, sip, smtp, snmp, sql-net, streamworks,

	sunrpc, sybase-sql, tacacs, telnet, tftp, or vdolive
port *port_num*	Identifies a port number in the range 1 to 65535.
list *acl_num*	Identifies the standard ACL number used with PAM for host- or network-specific port mapping.

Host- or Network-Specific Port Mapping

User-defined entries in the mapping table can include host- or network-specific mapping information, which establishes port mapping information for specific hosts or subnets. In some environments, it might be necessary to override the default port mapping information for a specific host or subnet.

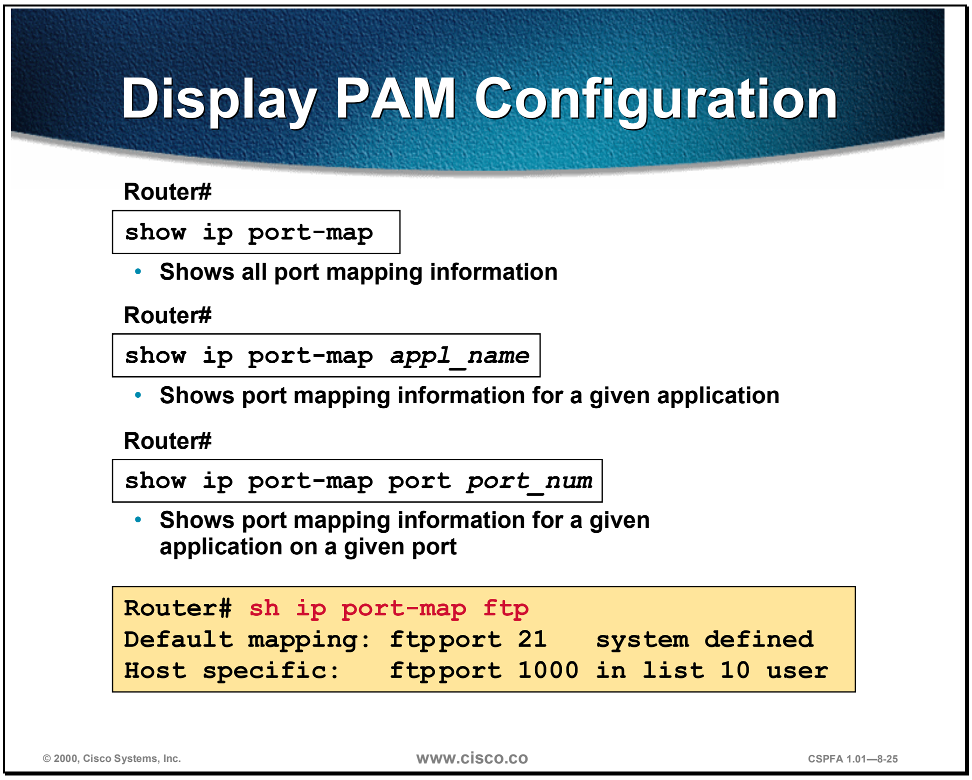
With host-specific port mapping, you can use the same port number for different services on different hosts. This means that you can map port 8000 with HTTP services for one host, while mapping port 8000 with Telnet services for another host.

Host-specific port mapping also allows you to apply PAM to a specific subnet when that subnet runs a service that uses a port number that is different from the port number defined in the default mapping information. For example, hosts on subnet 192.168.0.0 might run HTTP services on nonstandard port 8000, while other traffic through the firewall uses the default port 80 for HTTP services.

Host- or network-specific port mapping allows you to override a system-defined entry in the PAM table. For example, if CBAC finds an entry in the PAM table that maps port 25 (the system-defined port for SMTP) with HTTP for a specific host, CBAC identifies port 25 as HTTP protocol traffic on that host.

Note If the host-specific port mapping information is the same as existing system- or user-defined default entries, host-specific port changes have no effect.

Use the **list** option for the **ip port-map** command to specify an ACL for a host or subnet that uses PAM.



To display the PAM information, use the **show ip port-map** privileged EXEC command.

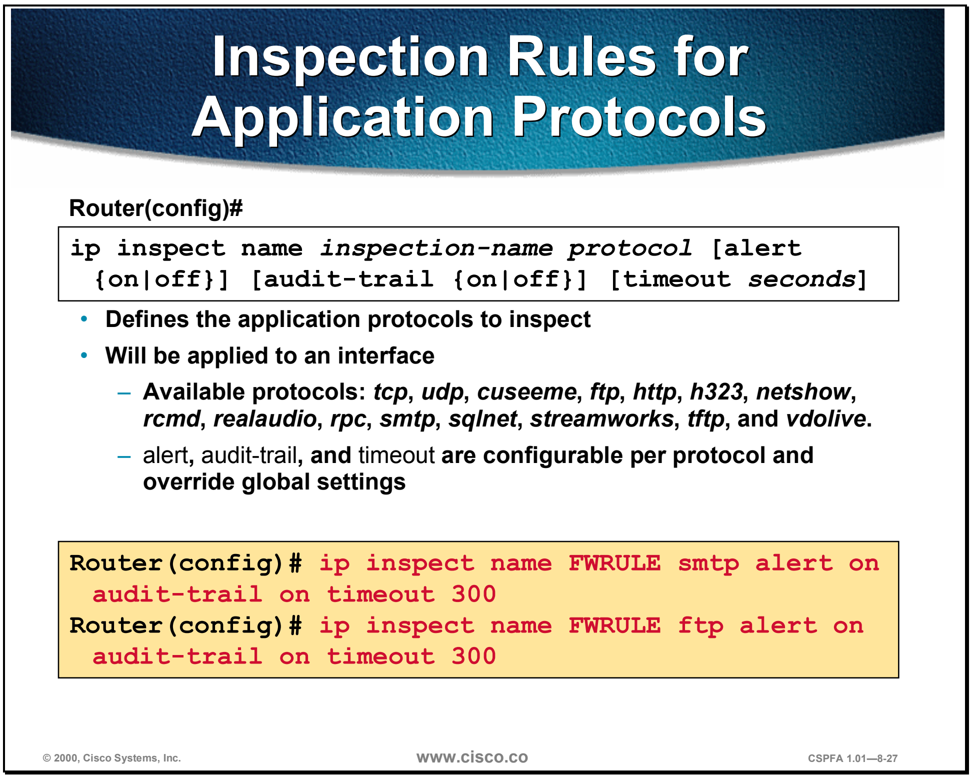
The syntax for the **show ip port-map** command is as follows:

show ip port-map [*appl_name* | port *port_num*]

Arguments	Description
appl_name	Specifies the application to display information for.
port *port_num*	Specifies the alternative port number that maps to the application to display information for.

Define Inspection Rules

This section discusses how to configure the rules used to define the application protocols for inspection.



Inspection rules must be defined to specify what IP traffic (which application-layer protocols) will be inspected by CBAC at an interface. Normally, you define only one inspection rule. The only exception might occur if you want to enable CBAC in two directions at a single firewall interface. In this case you must configure two rules, one for each direction.

An inspection rule should specify each desired application-layer protocol, as well as generic TCP or generic UDP, if desired. The inspection rule consists of a series of statements, each listing a protocol and specifying the same inspection rule name.

Inspection rules include options for controlling alert and audit trail messages and for checking IP packet fragmentation.

To define a set of inspection rules, use the **ip inspect name** command in global configuration mode. Use the **no** form of this command to remove the inspection rule for a protocol or to remove the entire set of inspection rules.

The syntax for the **ip inspect name** command is as follows:

ip inspect name ***inspection-name protocol*** **[alert {on | off}] [audit-trail {on | off}] [timeout** ***seconds*****]**

no ip inspect name ***inspection-name protocol***

no ip inspect name

Arguments	**Description**
name ***inspection-name***	Names the set of inspection rules. If you want to add a protocol to an existing set of rules, use the same ***inspection-name***.
protocol	The protocol to inspect. Use of the following keywords: **tcp**, **udp**, **cuseeme**, **ftp**, **http**, **h323**, **netshow**, **rcmd**, **realaudio**, **rpc**, **smtp**, **sqlnet**, **streamworks**, **tftp**, or **vdolive**.
alert {on | off}	(Optional.) For each inspected protocol, the generation of alert messages can be set on or off. If no option is selected, alerts are generated based on the setting of the **ip inspect alert-off** command.
audit-trail {on | **off}**	(Optional.) For each inspected protocol, **audit-trail** can be set on or off. If no option is selected, audit trail messages are generated based on the setting of the **ip inspect audit trail** command.
timeout ***seconds***	(Optional.) To override the global TCP or UDP idle timeouts for the specified protocol, specify the number of seconds for a different idle timeout. This timeout overrides the global TCP and UPD timeouts, but will not override the global DNS timeout.

Inspection Rules for Java

Router(config)#

```
ip inspect name inspection-name http java list
  acl-num [alert {on|off}] [audit-trail {on|off}]
  [timeout seconds]
```

- **Controls java blocking with standard access list**

```
Router(config)# ip inspect name FWRULE http java-list
  10 alert on audit-trail on timeout 300
Router(config)# ip access-list 10 deny 172.26.26.0
  0.0.0.255
Router(config)# ip access-list 10 permit 172.27.27.0
  0.0.0.255
```

 www.cisco.co CSPFA 1.01—8-28

Java inspection enables Java applet filtering at the firewall. Java applet filtering distinguishes between trusted and untrusted applets by relying on a list of external sites that you designate as friendly. If an applet is from a friendly site, the firewall allows the applet through. If the applet is not from a friendly site, the applet will be blocked. Alternately, you could permit applets from all sites except for sites specifically designated as hostile.

Note If you do not configure an access list, but use a "placeholder" access list in the **ip inspect name *inspection-name* http** command, all Java applets will be blocked.

Note CBAC does not detect or block encapsulated Java applets. Therefore, Java applets that are wrapped or encapsulated, such as applets in .zip or .jar format, are not blocked at the firewall. CBAC also does not detect or block applets loaded via FTP, gopher, or HTTP on a nonstandard port.

The syntax for the **ip inspect name** command for Java applet filtering inspection is as follows:

ip inspect name *inspection-name* http java-list *acl-num* [alert {on | off}] [audit-trail {on | off}] [timeout *seconds*]

no ip inspect name *inspection-name* http

Arguments	Description
name *inspection-name*	Names the set of inspection rules. If you want to add a protocol to an existing set of rules, use the same ***inspection-name*** as the existing set of rules.
http	Specifies the HTTP protocol for Java applet blocking.
java-list *acl-num*	Specifies the access list (name or number) to use to determine "friendly" sites. This keyword is available only for the HTTP protocol, for Java applet blocking. Java blocking only works with standard access lists.
alert {on | off}	(Optional.) For each inspected protocol, the generation of alert messages can be set on or off. If no option is selected, alerts are generated based on the setting of the **ip inspect alert-off** command.
audit-trail {on | off}	(Optional.) For each inspected protocol, **audit-trail** can be set on or off. If no option is selected, audit trail messages are generated based on the setting of the **ip inspect audit-trail** command.
timeout *seconds*	(Optional.) To override the global TCP or UDP idle timeouts for the specified protocol, specify the number of seconds for a different idle timeout. This timeout overrides the global TCP and UPD timeouts, but will not override the global DNS timeout.

Inspection Rules for RPC Applications

Router(config)#

```
ip inspect name inspection-name rpc
 program-number number [wait-time minutes]
 [alert {on|off}] [audit-trail {on|off}]
 [timeout seconds]
```

- **Allows given RPC program numbers**
  - wait-time **keeps the connection open for a specified number of minutes**

```
Router(config)# ip inspect name FWRULE rpc
 program-number 100022 wait-time 0 alert off
 audit-trail on
```

 www.cisco.co CSPFA 1.01—8-29

Remote Procedure Call (RPC) inspection allows the specification of various program numbers. You can define multiple program numbers by creating multiple entries for RPC inspection, each with a different program number. If a program number is specified, all traffic for that program number will be permitted. If a program number is not specified, all traffic for that program number will be blocked. For example, if you created an RPC entry with the NFS program number, all NFS traffic will be allowed through the firewall.

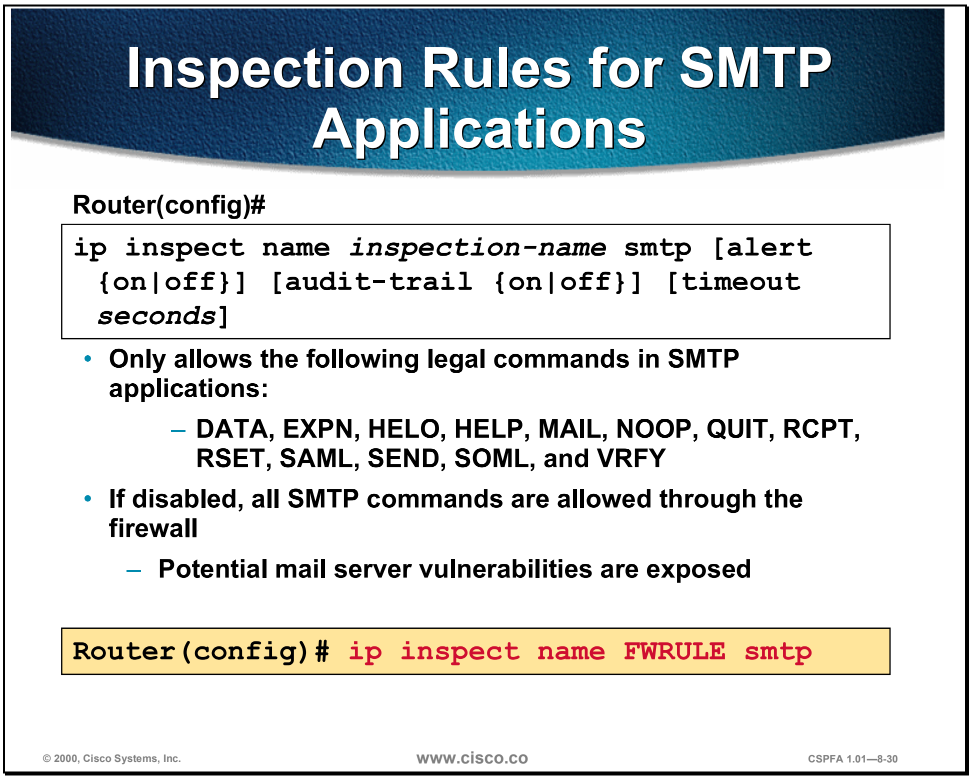
The syntax of the **ip inspect name** command for RPC applications is as follows:

ip inspect name ***inspection-name*** **rpc program-number** ***number*** **[wait-time** ***minutes*****] [alert** {**on** | **off**}**] [audit-trail** {**on** | **off**}**] [timeout** ***seconds*****]**

no ip inspect name ***inspection-name protocol***

Arguments	Description
inspection-name	Names the set of inspection rules. If you want to add a protocol to an existing set of rules, use the same ***inspection-name*** as the existing set of rules.
rpc program_number ***number***	Specifies the program number to permit.
wait-time *minutes*	(Optional.) Specifies the number of minutes to keep the connection opened in the firewall, even after the application terminates to allow subsequent connections from the same source address and to the same destination address and port. The default **wait-time** is zero minutes.
alert {on | off}	(Optional.) For each inspected protocol, the generation of alert messages can be set on or off. If no option is selected, alerts are generated based on the setting of the **ip inspect alert-off** command.

audit-trail {on | **off}**	(Optional.) For each inspected protocol, **audit-trail** can be set on or off. If no option is selected, audit trail messages are generated based on the setting of the **ip inspect audit-trail** command.
timeout *seconds*	(Optional.) To override the global TCP or UDP idle timeouts for the specified protocol, specify the number of seconds for a different idle timeout. This timeout overrides the global TCP and UPD timeouts, but will not override the global DNS timeout.



SMTP inspection causes SMTP commands to be inspected for illegal commands. Any packets with illegal commands are dropped, and the SMTP session hangs and eventually times out. An illegal command is any command except for the following legal commands: DATA, EXPN, HELO, HELP, MAIL, NOOP, QUIT, RCPT, RSET, SAML, SEND, SOML, and VRFY.

The syntax for the **ip inspect name** command for SMTP application inspection is as follows:

ip inspect name ***inspection-name*** **smtp [alert** {**on** | **off**}**] [audit-trail** {**on** | **off**}**] [timeout** ***seconds*****]**

no ip inspect name ***inspection-name*** **smtp**

Arguments	Description
name *inspection-name*	Names the set of inspection rules. If you want to add a protocol to an existing set of rules, use the same ***inspection-name*** as the existing set of rules.
smtp	Specifies the SMTP protocol for inspection.
alert {on | off}	(Optional.) For each inspected protocol, the generation of alert messages can be set on or off. If no option is selected, alerts are generated based on the setting of the **ip inspect alert-off** command.
audit-trail {on | off}	(Optional.) For each inspected protocol, **audit-trail** can be set on or off. If no option is selected, audit trail messages are generated based on the setting of the **ip inspect audit-trail** command.
timeout *seconds*	(Optional.) To override the global TCP or UDP idle timeouts for the specified protocol, specify the number of seconds for a different idle timeout. This timeout overrides the global TCP and UPD timeouts, but will not override the global DNS timeout.

Inspection Rules for IP Packet Fragmentation

Router(config)#

```
ip inspect name inspection-name fragment max
  number timeout seconds
```

- **Protects hosts from certain DoS attacks involving fragmented IP packets**
  - **max = number of unassembled fragmented IP packets**
  - **timeout = seconds when the unassembled fragmented IP packets begin to be discarded**

```
Router(config)# ip inspect name FWRULE
  fragment max 254 timeout 4
```

 www.cisco.co CSPFA 1.01—8-31

CBAC inspection rules can help protect hosts against certain DoS attacks involving fragmented IP packets. Even though the firewall keeps an attacker from making actual connections to a given host, the attacker may still be able to disrupt services provided by that host. This is done by sending many noninitial IP fragments, or by sending complete fragmented packets through a router with an ACL that filters the first fragment of a fragmented packet. These fragments can tie up resources on the target host as it tries to reassemble the incomplete packets.

Using fragmentation inspection, the firewall maintains an interfragment state (structure) for IP traffic. Noninitial fragments are discarded unless the corresponding initial fragment was permitted to pass through the firewall. Noninitial fragments received before the corresponding initial fragments are discarded.

Note Fragmentation inspection can have undesirable effects in certain cases, because it can result in the firewall discarding any packet whose fragments arrive out of order. There are many circumstances that can cause out-of-order delivery of legitimate fragments. Apply fragmentation inspection in situations where legitimate fragments, which are likely to arrive out of order, might have a severe performance impact.

Because routers running Cisco IOS software are used in a very large variety of networks, and because the CBAC feature is often used to isolate parts of internal networks from one another, the fragmentation inspection feature is not enabled by default. Fragmentation detection must be explicitly enabled for an inspection rule using the **ip inspect name** (global) command. Unfragmented traffic is never discarded because it lacks a fragment state. Even when the system is under heavy attack with fragmented packets, legitimate fragmented traffic, if any, will still get some fraction of the firewall's fragment state resources, and legitimate, unfragmented traffic can flow through the firewall unimpeded.

The syntax of the **ip inspect name** command for IP packet fragmentation is as follows:

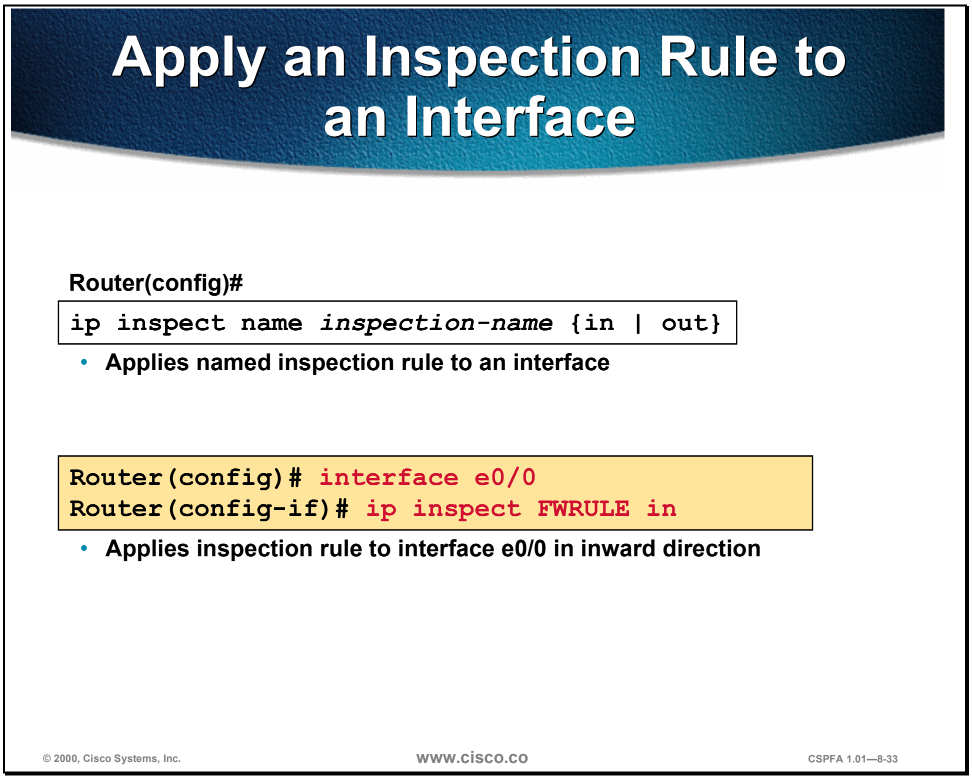
ip inspect name *inspection-name* **fragment max** *number* **timeout** *seconds*

no ip inspect name *inspection-name* **fragment**

Arguments	Description
inspection-name	Names the set of inspection rules. If you want to add a protocol to an existing set of rules, use the same ***inspection-name*** as the existing set of rules.
fragment	Specifies fragment inspection for the named rule.
max ***number***	Specifies the maximum number of unassembled packets for which state information (structures) is allocated by the software. Unassembled packets are packets that arrive at the router interface before the initial packet for a session. The acceptable range is 50 through 10000. The default is 256 state entries. Memory is allocated for the state structures, and setting this value to a larger number may cause memory resources to be exhausted.
timeout ***seconds***	Configures the number of seconds that a packet state structure remains active. When the timeout value expires, the router drops the unassembled packet, freeing that structure for use by another packet. The default timeout value is one second. If this number is set to a value greater than one second, it will be automatically adjusted by the software when the number of free state structures goes below certain thresholds: when the number of free states is less than 32, the timeout will be divided by 2. When the number of free states is less than 16, the timeout will be set to 1 second.

Inspection Rules and ACLs Applied to Router Interfaces

This section discusses the application of inspection rules and ACLs to router interfaces.



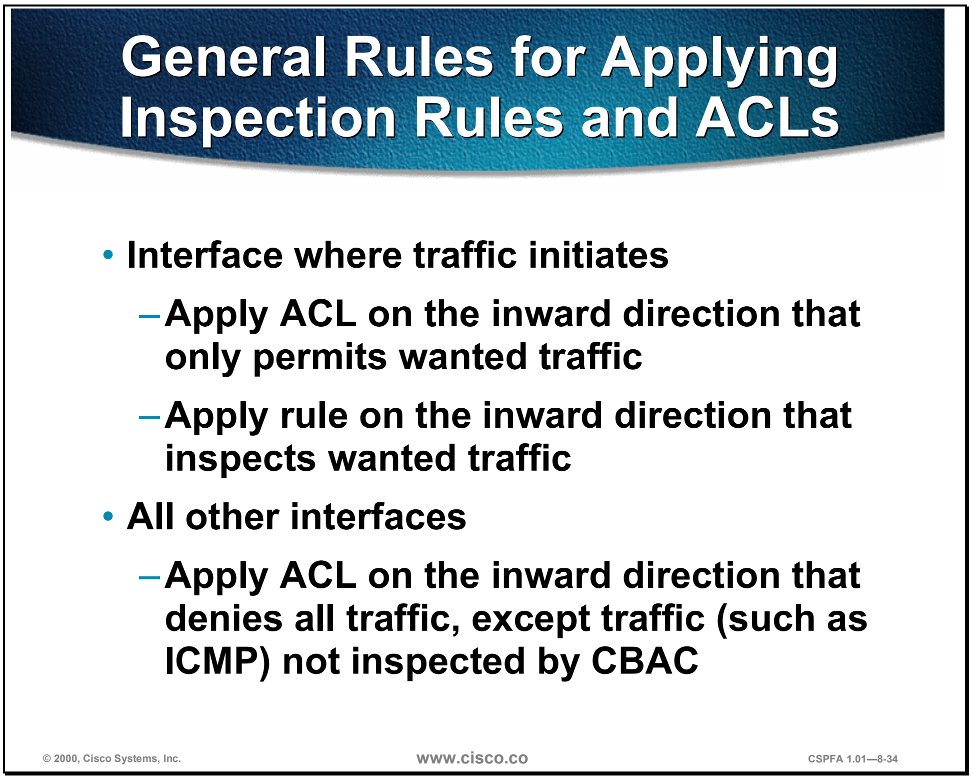
To apply a set of inspection rules to an interface, use the **ip inspect** interface configuration command. Use the **no** form of this command to remove the set of rules from the interface.

The syntax for the **ip inspect** command is as follows:

ip inspect name ***inspection-name*** **{in | out }**

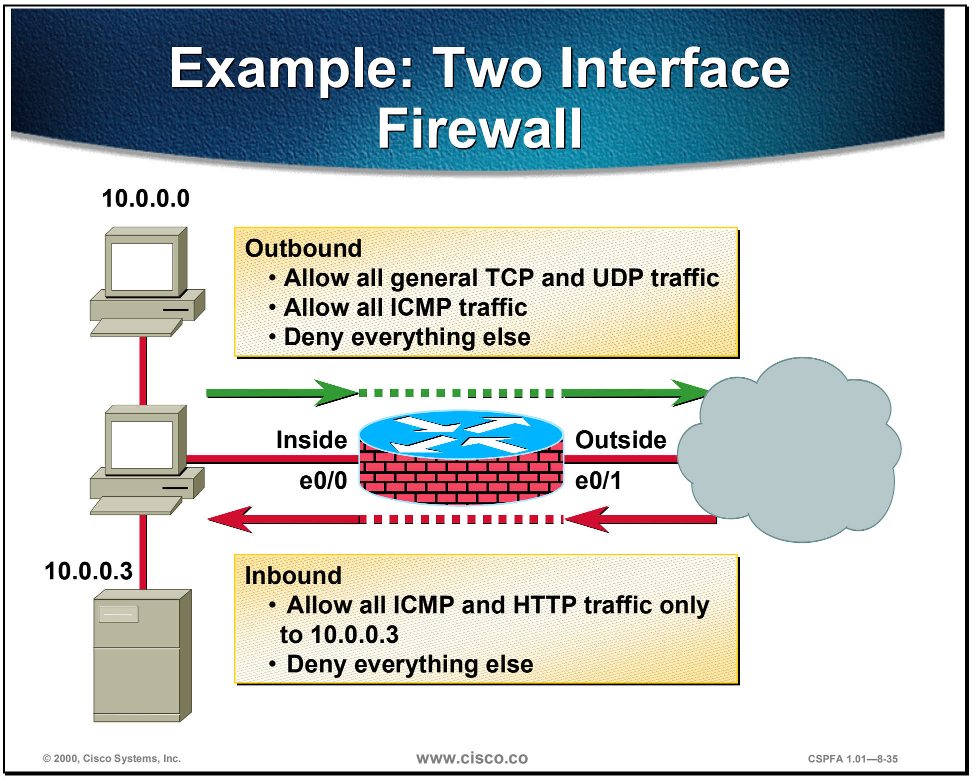
no ip inspect inspection-name {in | out}

Arguments	Description
inspection-name	Names the set of inspection rules.
in	Applies the inspection rules to inbound traffic.
out	Applies the inspection rules to outbound traffic.

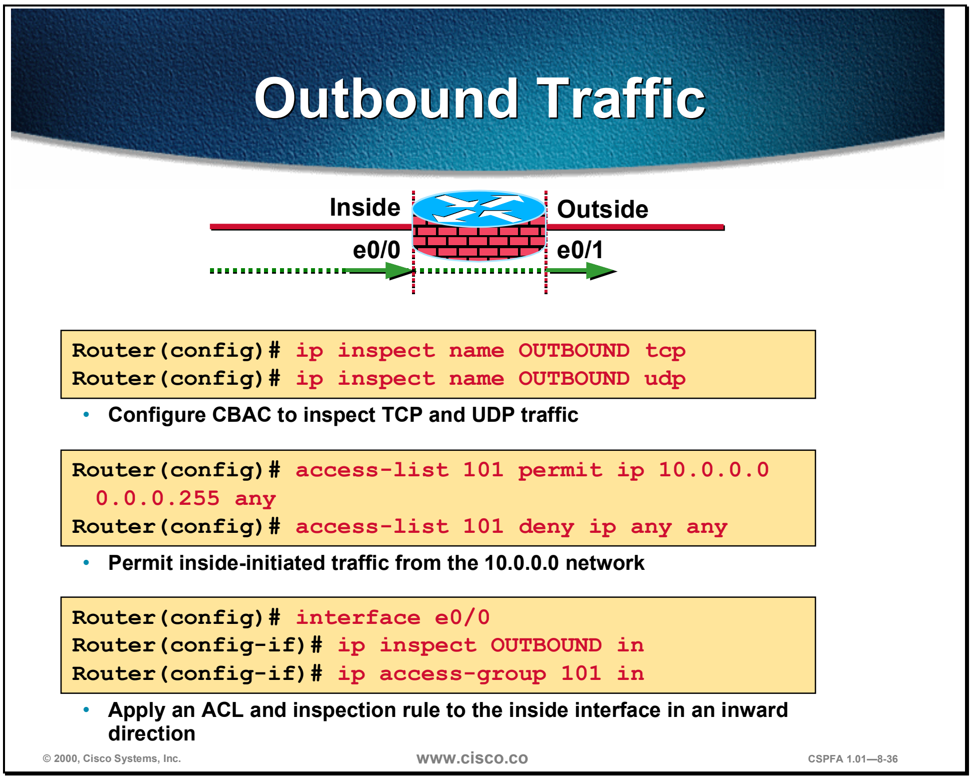


For the CISCO IOS Firewall to be effective, both inspection rules and ACLs must be strategically applied to all the router's interfaces. The following is the general rule of thumb for applying inspection rules and ACLs on the router:

- On the interface where traffic initiates
  - Apply the ACL on the inward direction that only permits wanted traffic.
  - Apply the rule on the inward direction that inspects wanted traffic.
- All other interfaces
  - Apply the ACL on the inward direction that denies all traffic, except traffic (such as ICMP) not inspected by CBAC.



As an example, configure the router to be a firewall between two networks: inside and outside. The security policy to implement is as follows: allow all general TCP and UDP traffic initiated on the inside (outbound) from network 10.0.0.0 to access the Internet. ICMP traffic will also be allowed from the same network. Other networks on the inside, which are not defined, must be denied. For traffic initiated on the outside (inbound), allow everyone to only access ICMP and HTTP to host 10.0.0.3. Any other traffic must be denied.



To implement the security policy of the previous example, do the following for outbound traffic:

Step 1 Write a rule to inspect TCP and UDP traffic:

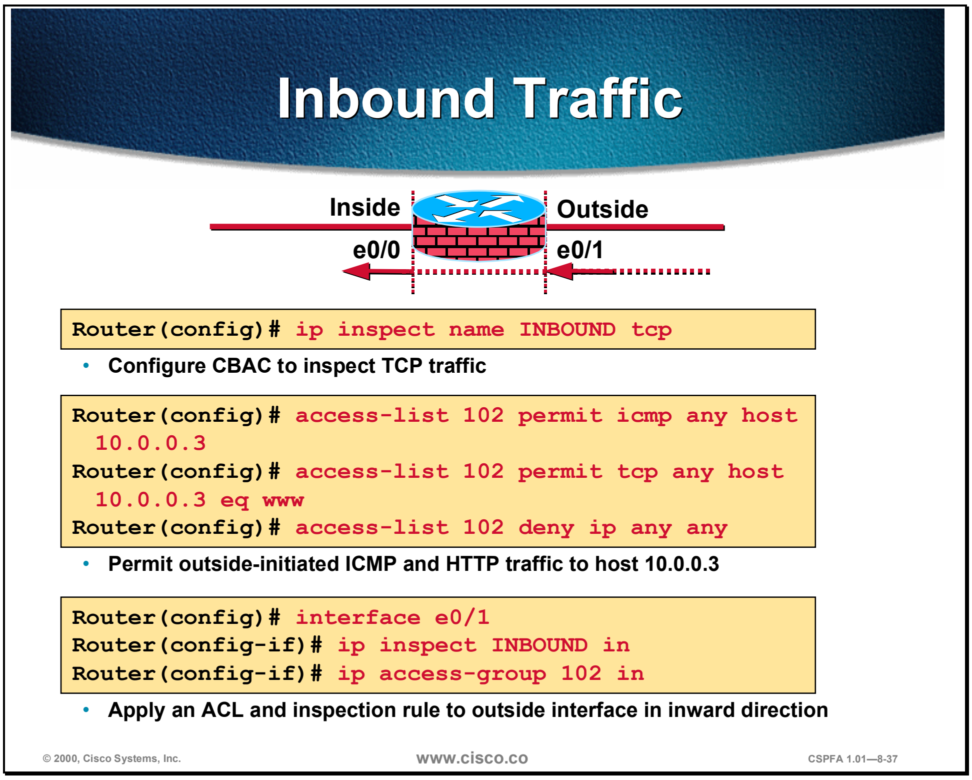
```
Router(config)# ip inspect name OUTBOUND tcp
Router(config)# ip inspect name OUTBOUND udp
```

Step 2 Write an ACL that permits IP traffic from the 10.0.0.0 network to any destination:

```
Router(config)# access-list 101 permit ip 10.0.0.0 0.0.0.255 any
Router(config)# access-list 101 deny ip any any
```

Step 3 Apply the inspection rule and ACL to the inside interface on the inward direction:

```
Router(config)# interface e0/0
Router(config-if)# ip inspect OUTBOUND in
Router(config-if)# ip access-group 101 in
```



To implement the security policy of the previous example, do the following for inbound traffic:

Step 1 Write a rule to inspect TCP traffic:

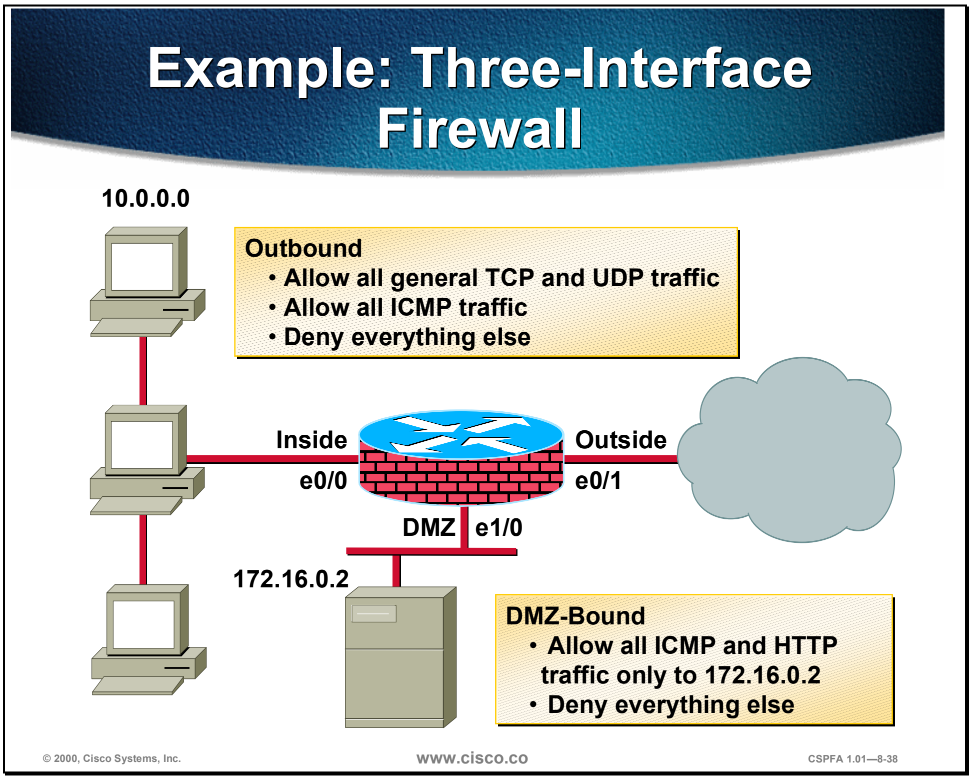
```
Router(config)# ip inspect name INBOUND tcp
```

Step 2 Write an ACL that permits ICMP and HTTP-only traffic from the Internet to the 10.0.0.3 host:

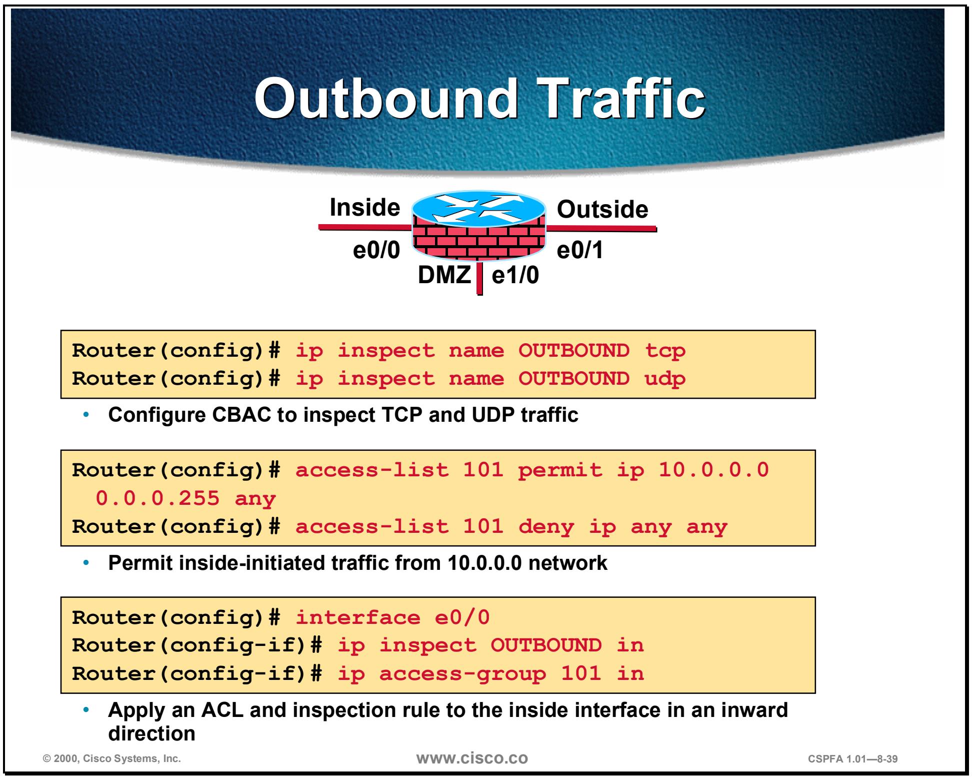
```
Router(config)# access-list 102 permit icmp any host 10.0.0.3
Router(config)# access-list 102 permit tcp any host 10.0.0.3 eq www
Router(config)# access-list 102 deny ip any any
```

Step 3 Apply the inspection rule and ACL to the outside interface in the inward direction:

```
Router(config)# interface e0/1
Router(config-if)# ip inspect INBOUND in
Router(config-if)# ip access-group 102 in
```

As an example, configure the router to be a firewall between three networks: inside, outside, and DMZ. The security policy to implement is as follows: allow all general TCP and UDP traffic initiated on the inside (outbound) from network 10.0.0.0 to access the Internet and the DMZ host 172.16.0.2. ICMP traffic will also be allowed from the same network to the Internet and the DMZ host. Other networks on the inside, which are not defined, must be denied. For traffic initiated on the outside (inbound) allow everyone to only access ICMP and HTTP to DMZ host 172.16.0.2. Any other traffic must be denied.



To implement the security policy of the previous example, do the following for outbound traffic:

Step 1 Write a rule to inspect TCP and UDP traffic:

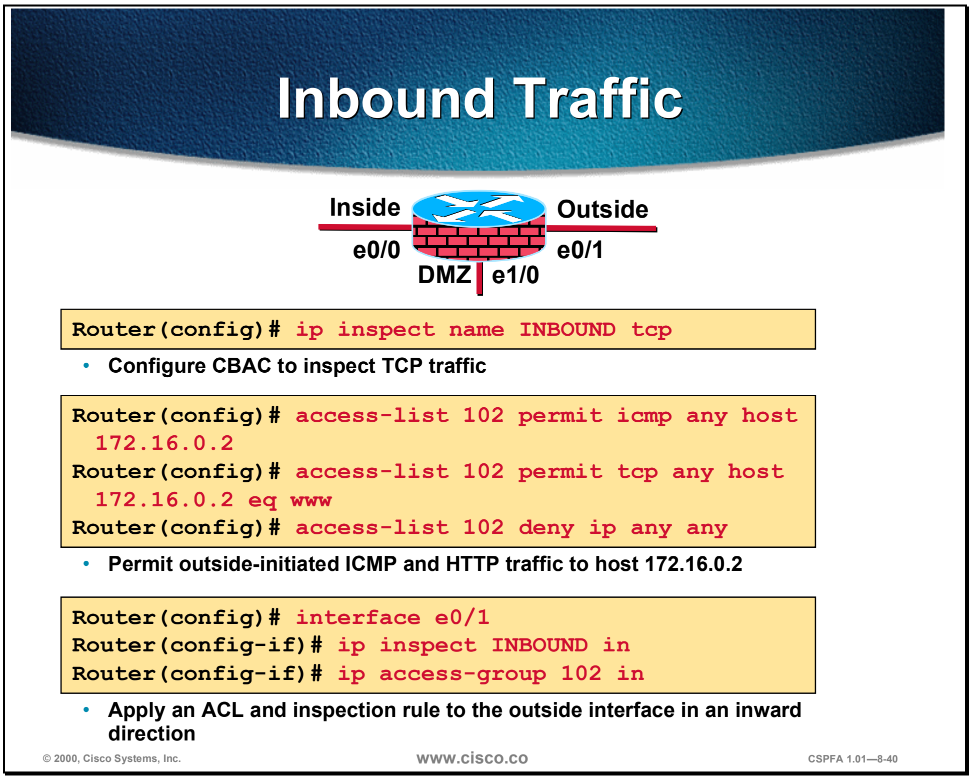
```
Router(config)# ip inspect name OUTBOUND tcp
Router(config)# ip inspect name OUTBOUND udp
```

Step 2 Write an ACL that permits IP traffic from the 10.0.0.0 network to any destination:

```
Router(config)# access-list 101 permit ip 10.0.0.0 0.0.0.255 any
Router(config)# access-list 101 deny ip any any
```

Step 3 Apply the inspection rule and ACL to the inside interface in the inward direction:

```
Router(config)# interface e0/0
Router(config-if)# ip inspect OUTBOUND in
Router(config-if)# ip access-group 101 in
```



To implement the security policy of the previous example, do the following for inbound traffic:

Step 1 Write a rule to inspect TCP traffic:

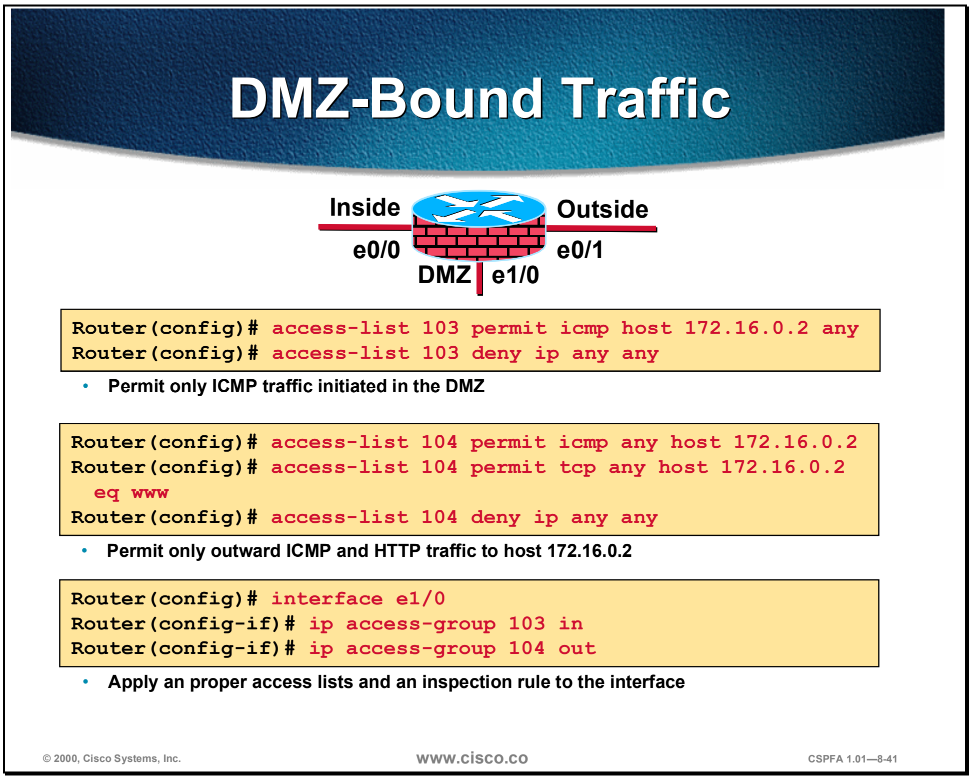
```
Router(config)# ip inspect name INBOUND tcp
```

Step 2 Write an ACL that permits ICMP and HTTP-only traffic from the Internet to the 172.16.0.2 host:

```
Router(config)# access-list 102 permit icmp any host 172.16.0.2
Router(config)# access-list 102 permit tcp any host 172.16.0.2 eq www
Router(config)# access-list 102 deny ip any any
```

Step 3 Apply the inspection rule and ACL to the outside interface in the inward direction:

```
Router(config)# interface e0/1
Router(config-if)# ip inspect INBOUND in
Router(config-if)# ip access-group 102 in
```



To implement the security policy of the previous example, do the following for inbound traffic:

Step 1 Write an ACL to permit only ICMP traffic to initiate from the DMZ host:

```
Router(config)# access-list 103 permit icmp host 172.16.0.2 any
Router(config)# access-list 103 deny ip any any
```

Step 2 Write an ACL that permits ICMP and HTTP-only traffic from any network to the 172.16.0.2 host:

```
Router(config)# access-list 104 permit icmp any host 172.16.0.2
Router(config)# access-list 104 permit tcp any host 172.16.0.2 eq www
Router(config)# access-list 104 deny ip any any
```

Step 3 Apply the ACLs to the DMZ interface:

```
Router(config)# interface e1/0
Router(config-if)# ip access-group 103 in
Router(config-if)# ip access-group 104 out
```


Test and Verify

This section discusses the commands available to help test and verify CBAC.

show Commands

Router#

```
show ip inspect name inspection-name
show ip inspect config
show ip inspect interfaces
show ip inspect session [detail]
show ip inspect all
```

- Displays CBAC configurations, interface configurations, and sessions

```
Router# sh ip inspect session
Established Sessions
 Session 6155930C (10.0.0.3:35009)=>(172.30.0.50:34233)
  tcp SIS_OPEN
 Session 6156F0CC (10.0.0.3:35011)=>(172.30.0.50:34234)
  tcp SIS_OPEN
 Session 6156AF74 (10.0.0.3:35010)=>(172.30.0.50:5002) tcp
  SIS_OPEN
```

 www.cisco.co CSPFA 1.01—8-43

The syntax for the **show ip inspect** command is as follows:

show ip inspect name ***inspection-name*** | **config** | **interfaces** | **session [detail]** | **all**

Arguments	Description
inspection-name	Shows the configured inspection rule for ***inspection-name***.
config	Shows the complete CBAC inspection configuration.
interfaces	Shows interface configuration with respect to applied inspection rules and access lists.
session [detail]	Shows existing sessions that are currently being tracked and inspected by CBAC. The optional **detail** keyword shows additional details about these sessions.
all	Shows the complete CBAC configuration and all existing sessions that are currently being tracked and inspected by CBAC.

debug Commands

Router#

```
debug ip inspect function-trace
debug ip inspect object-creation
debug ip inspect object-deletion
debug ip inspect events
debug ip inspect timers
```

- **General debug commands**

Router(config)#

```
debug ip inspect protocol
```

- **Protocol-specific debug**

 www.cisco.co CSPFA 1.01—8-44

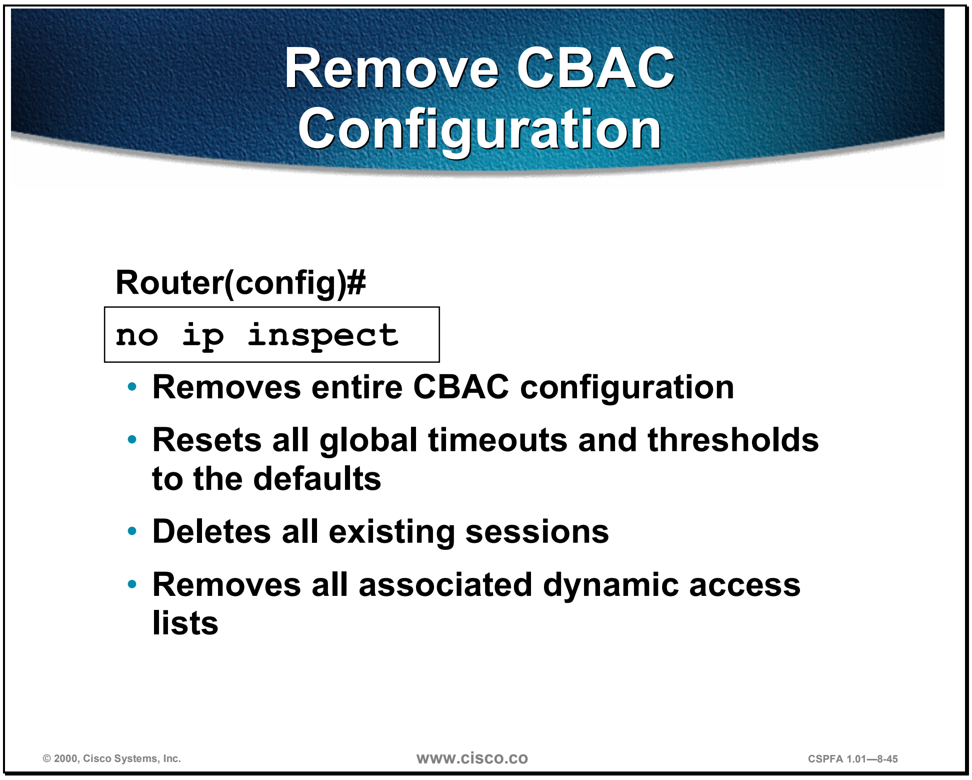
To display messages about CBAC events, use the **debug ip inspect** EXEC command. The **no** form of this command disables debugging output.

The syntax for the **debug ip inspect** command is as follows:

debug ip inspect {function-trace | object-creation | object-deletion | events | timers | ***protocol*** **| detailed}**

no debug ip inspect

Arguments	Description
function-trace	Displays messages about software functions called by CBAC.
object-creation	Displays messages about software objects being created by CBAC. Object creation corresponds to the beginning of CBAC-inspected sessions.
object-deletion	Displays messages about software objects being deleted by CBAC. Object deletion corresponds to the closing of CBAC-inspected sessions.
events	Displays messages about CBAC software events, including information about CBAC packet processing.
timers	Displays messages about CBAC timer events, such as when a CBAC idle timeout is reached.
protocol	Displays messages about CBAC-inspected protocol events, including details about the protocol's packets.
detailed	Use this form of the command in conjunction with other CBAC debugging commands. This displays detailed information for all other enabled CBAC debugging.



Use the **no ip inspect** command to remove the entire CBAC configuration, reset all global timeouts and thresholds to their defaults, delete all existing sessions, and remove all associated dynamic access lists. This command has no other arguments, keywords, default behavior, or values.

Lab Exercise: Configure CBAC on a Cisco Router

Complete the following lab exercise to practice what you have learned in this chapter.

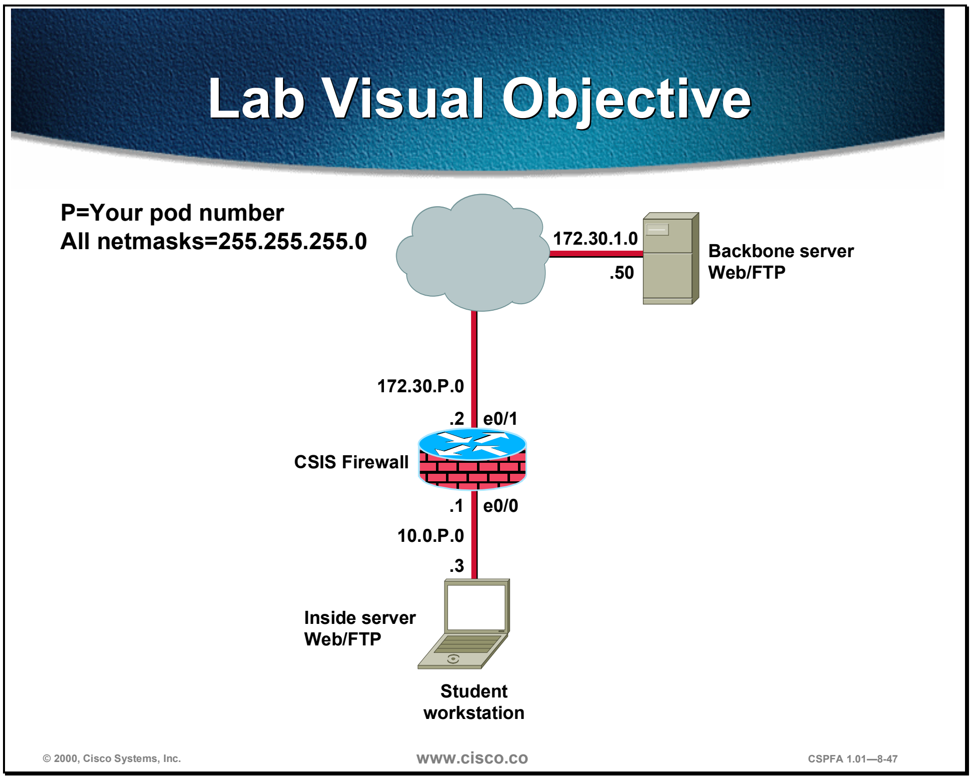
Objectives

In this lab you will complete the following tasks:

- Configure basic router security.
- Configure logging and audit trails.
- Define and apply inspection rules and access lists.
- Test and verify CBAC.

Lab Visual Objective

The following figure displays the pod configuration that you will use to complete this lab exercise.



Task 1: Configure Logging and Audit Trails

Step 1 On your workstation, start the syslog server by choosing **Start>Programs>Syslogd** from the menu bar.

Step 2 On your router, enable logging to the console and the syslog server:

```
Router(config)# logging on
Router(config)# logging 10.0.P.3
```

(where P = pod number)

Step 3 Enable audit trail:

```
Router(config)# ip inspect audit-trail
```

Step 4 Save your configuration and return to global configuration mode:

```
Router(config)# end
Router# write memory
```

Task 2: Define and Apply Inspection Rules and Access Lists

Step 1 On your router, define a CBAC rule to inspect all TCP and FTP traffic:

```
Router(config)# ip inspect name FWRULE tcp timeout 300
Router(config)# ip inspect name FWRULE ftp timeout 300
```

Step 2 Define **access-list** to allow outbound ICMP traffic and CBAC traffic (FTP and WWW). Block all other inside-initiated traffic:

```
Router(config)# access-list 101 permit icmp any any
Router(config)# access-list 101 permit tcp 10.0.P.0 0.0.0.255 any eq ftp
```

```
Router(config)# access-list 101 permit tcp 10.0.P.0 0.0.0.255 any eq www
Router(config)# access-list 101 deny ip any any
```

(where P = pod number)

Step 3 Define access-list to allow inbound ICMP traffic and CBAC traffic (FTP and WWW) to the inside web/FTP server. Block all other outside-initiated traffic:

```
Router(config)# access-list 102 permit eigrp any any
Router(config)# access-list 102 permit icmp any any
Router(config)# access-list 102 permit tcp any host 10.0.P.3 eq ftp
Router(config)# access-list 102 permit tcp any host 10.0.P.3 eq www
Router(config)# access-list 102 deny ip any any
```

(where P = pod number)

Step 4 Apply the inspection rule and access list to the inside interface:

```
Router(config)# interface ethernet 0/0
Router(config-if)# ip inspect FWRULE in
Router(config-if)# ip access-group 101 in
```

Step 5 Apply the access list to the outside interface:

```
Router(config-if)# interface ethernet 0/1
Router(config-if)# ip inspect FWRULE in
Router(config-if)# ip access-group 102 in
```

Step 6 Save your configuration and return to global configuration mode:

```
Router(config-if)# end
Router# write memory
```

Task 3: Test and Verify CBAC

Step 1 Check your access lists:

```
Router# show access-lists
```

Step 2 From your workstation command prompt, ping the backbone server:

```
C:\> ping 172.30.1.50
Pinging 172.30.0.50 with 32 bytes of data:

Reply from 172.30.1.50: bytes=32 time=34ms TTL=125
Reply from 172.30.1.50: bytes=32 time=34ms TTL=125
Reply from 172.30.1.50: bytes=32 time=34ms TTL=125
Reply from 172.30.1.50: bytes=32 time=36ms TTL=125
```

Step 3 Use your Web browser to connect to the backbone Web server. Enter **http://172.30.1.50** in the URL field.

Step 4 Connect to the backbone FTP server using anonymous FTP:

```
C:\> ftp 172.30.1.50
...
User (10.0.0.3:(none)): anonymous
...
Password: user@
```

Step 5 Do a directory listing to verify data channel connectivity:


```
ftp> ls
```

Step 6 On your router, use the following **show** commands to verify CBAC operation:

```
Router# show ip inspect name FWRULE
Router# show ip inspect config
Router# show ip inspect interfaces
Router# show ip inspect sessions
Router# show ip inspect sessions detail
Router# show ip inspect all
```

Step 7 From your workstation command prompt, ping your peer's inside server:

```
C:\> ping 10.0.Q.3
Pinging 10.0.1.3 with 32 bytes of data:

Reply from 10.0.1.3: bytes=32 time=34ms TTL=125
Reply from 10.0.1.3: bytes=32 time=34ms TTL=125
Reply from 10.0.1.3: bytes=32 time=34ms TTL=125
Reply from 10.0.1.3: bytes=32 time=36ms TTL=125
```

Use your Web browser to connect to your peer's inside server. Enter **http://10.0.Q.3** in the URL field

(where Q = peer pod number)

Step 8 Connect to your peer's FTP server using anonymous FTP:

```
C:\> ftp 10.0.Q.3
...
User (10.0.1.3:(none)): anonymous
...
Password: user@
```

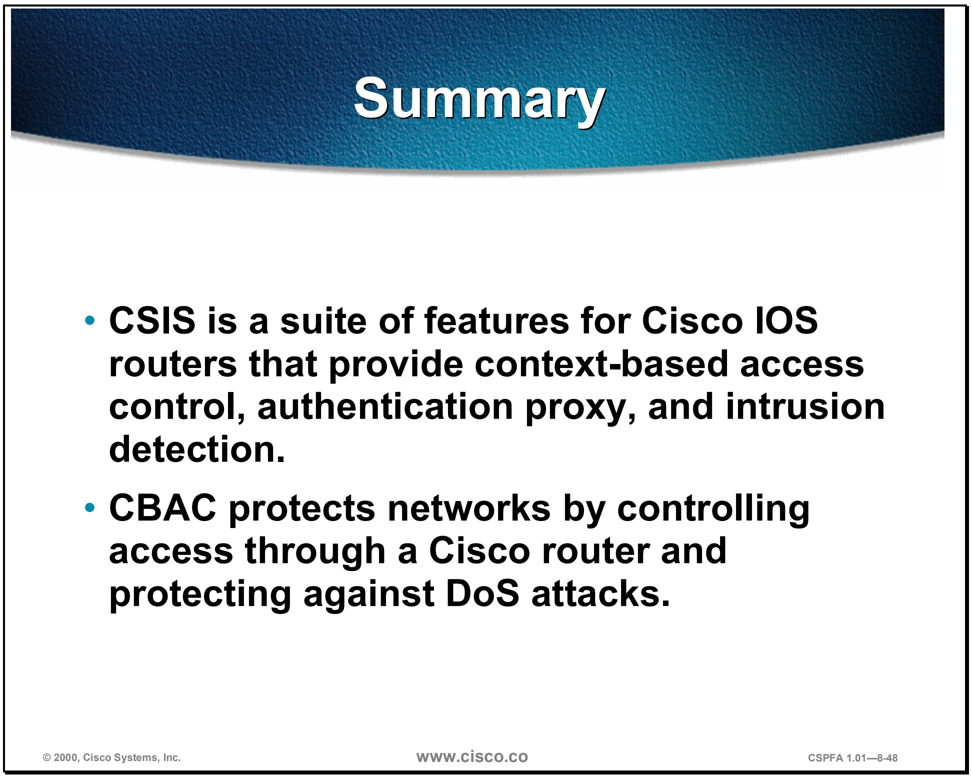
(where Q = peer pod number)

Step 9 On your router, use the following **show** commands to verify CBAC operation:

```
Router# show ip inspect name FWRULE
Router# show ip inspect config
Router# show ip inspect interfaces
Router# show ip inspect sessions
Router# show ip inspect sessions detail
Router# show ip inspect all
```

Summary

This section summarizes what you learned in this chapter.



9

Cisco IOS Firewall Authentication Proxy Configuration

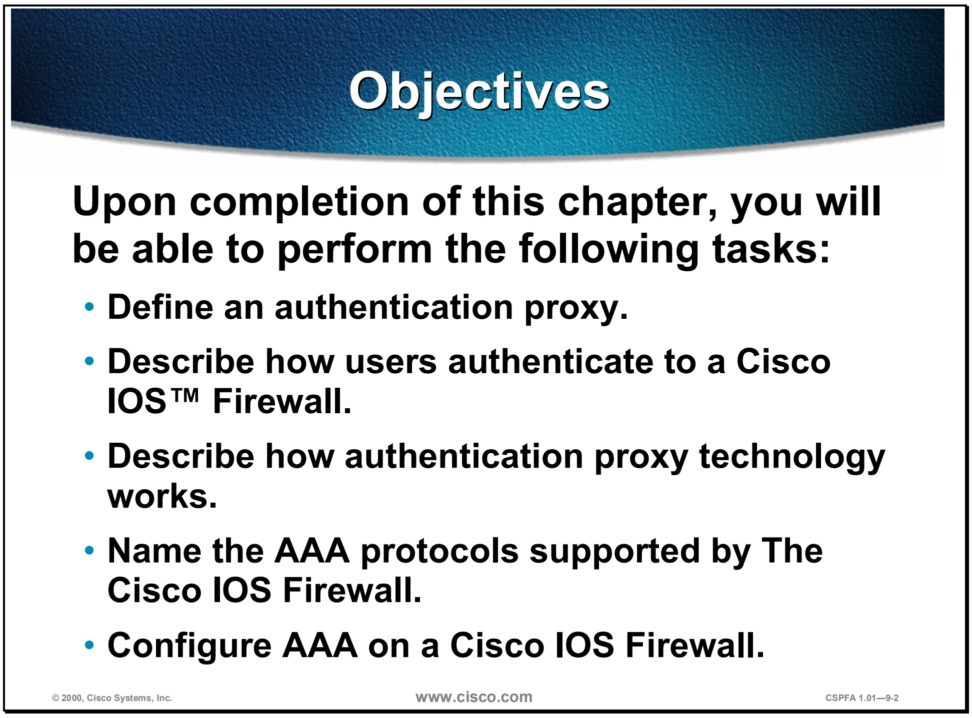
Overview

This chapter includes the following topics:

- Introduction to the Cisco IOS authorization proxy
- AAA server configuration
- AAA configuration
- Authentication proxy configuration
- Test and verify the configuration
- Lab exercise
- Summary

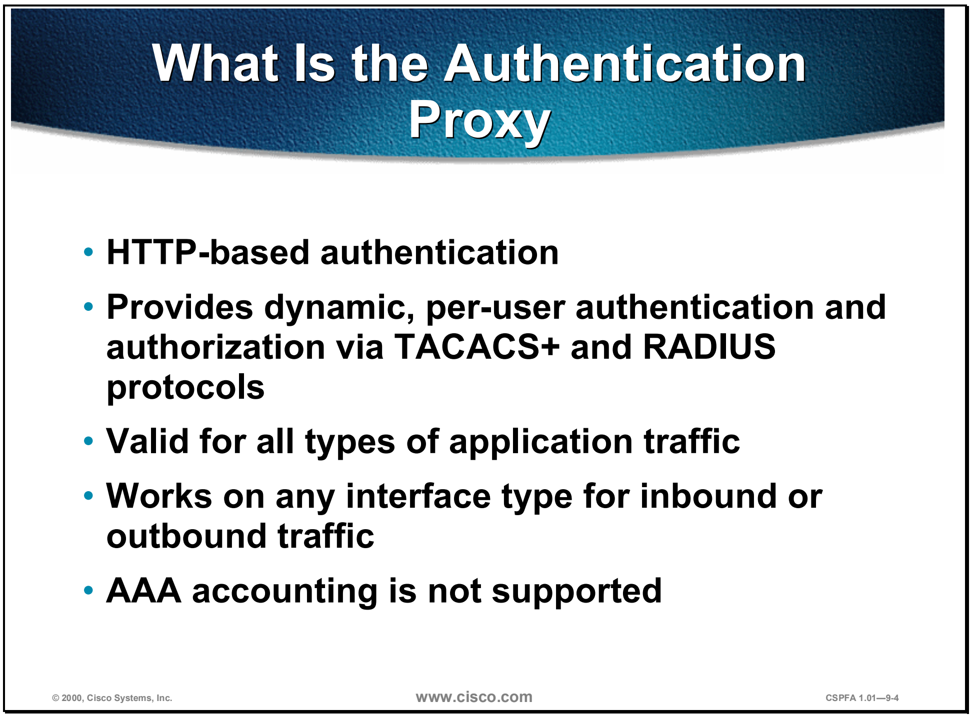
Objectives

This section lists the chapter's objectives.



Introduction to the IOS Authorization Proxy

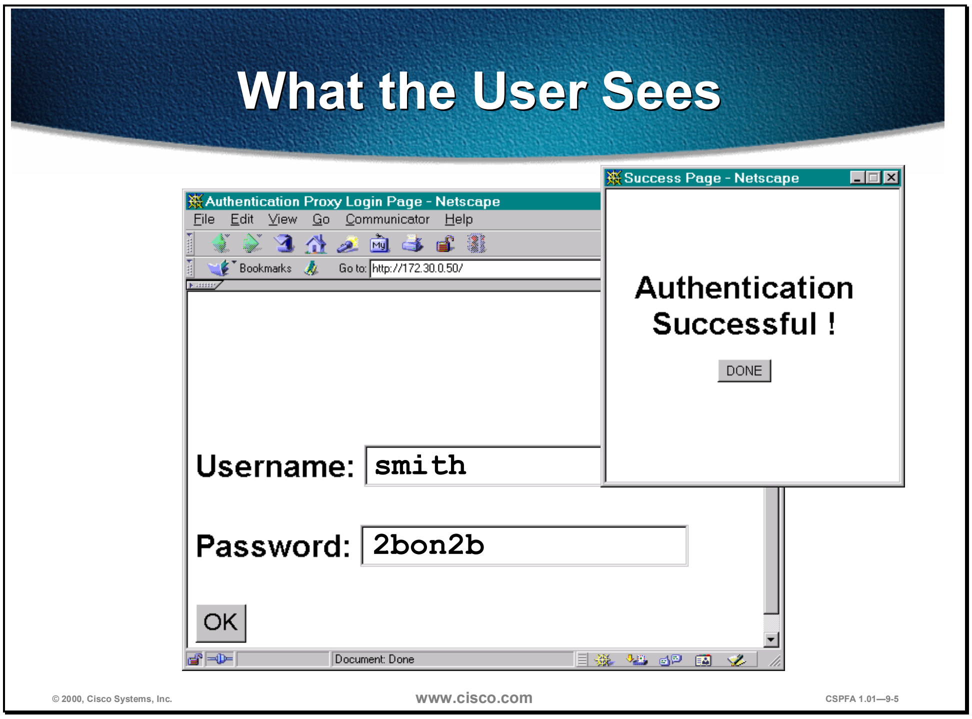
This section introduces the features of the IOS™ Firewall authorization proxy.



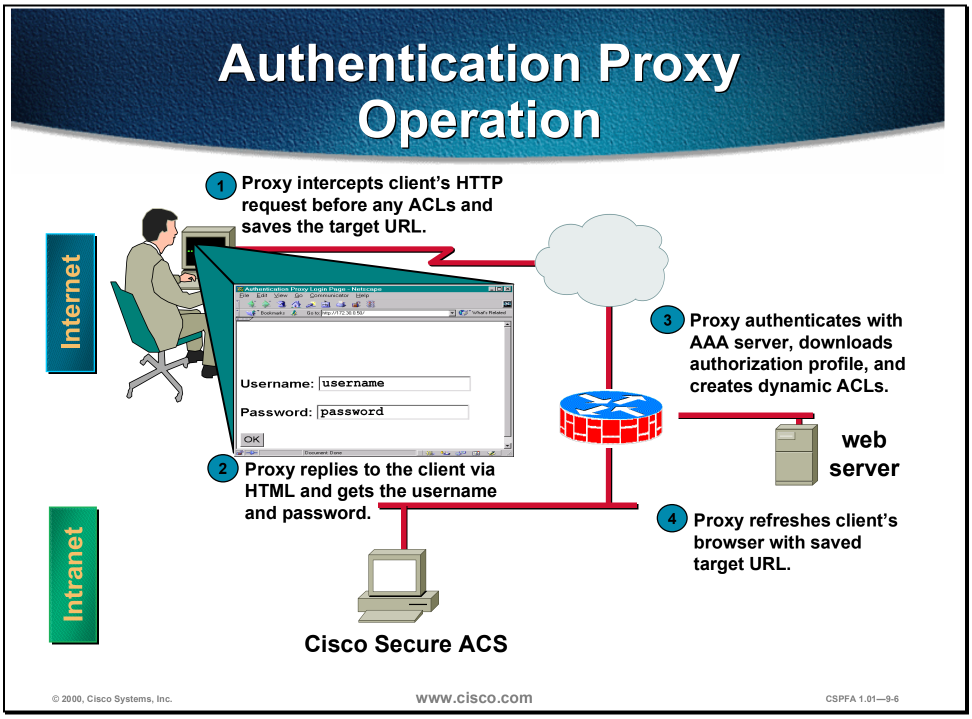
The Cisco IOS Firewall authentication proxy feature allows network administrators to apply specific security policies on a per-user basis. Previously, user identity and related authorized access were associated with a user's IP address, or a single security policy had to be applied to an entire user group or subnet. Now, users can be identified and authorized on the basis of their per-user policy, and access privileges can be tailored on an individual basis, as opposed to a general policy applied across multiple users.

With the authentication proxy feature, users can log into the network or access the Internet via HTTP, and their specific access profiles are automatically retrieved and applied from a Cisco Secure Asynchronous Communication Server (ACS), or other RADIUS or TACACS+ authentication server. The user profiles are active only when there is active traffic from the authenticated users.

The authentication proxy is compatible with other Cisco IOS security features such as Network Address Translation (NAT), Context-Based Access Control (CBAC), IP Security (IPSec) encryption, and VPN client software.



When a user initiates an HTTP session through the firewall, it triggers the authentication proxy. If a valid authentication entry exists for the user, the session is allowed and no further intervention is required by the authentication proxy. If no entry exists, the authentication proxy responds to the HTTP connection request by prompting the user for a username and password, as shown above.



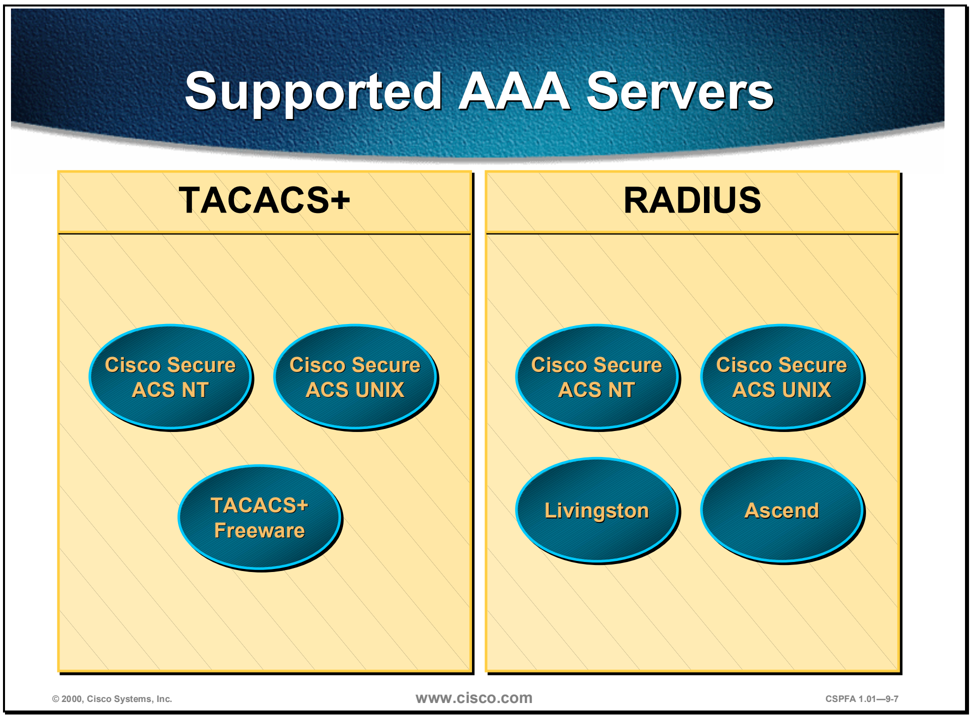
When a user initiates an HTTP session through the firewall, it triggers the authentication proxy. The authentication proxy first checks to see if the user has been authenticated. If a valid authentication entry exists for the user, the session is allowed and no further intervention is required by the authentication proxy. If no entry exists, the authentication proxy responds to the HTTP connection request by prompting the user for a username and password.

Users must successfully authenticate with the authentication server by entering a valid username and password. If the authentication succeeds, the user's authorization profile is retrieved from the authentication, authorization, and accounting (AAA) server. The authentication proxy uses the information in the this profile to create dynamic access control entries (ACEs) and add them to the inbound (input) access control list (ACL) of an input interface, and to the outbound (output) ACL of an output interface if an output ACL exists at the interface. By doing this, the firewall allows authenticated users access to the network as permitted by the authorization profile. For example, a user can initiate a Telnet connection through the firewall if Telnet is permitted in the user's profile.

If the authentication fails, the authentication proxy reports the failure to the user, and prompts the user with multiple retries. If the user fails to authenticate after five attempts, the user must wait two minutes and initiate another HTTP session to trigger the authentication proxy.

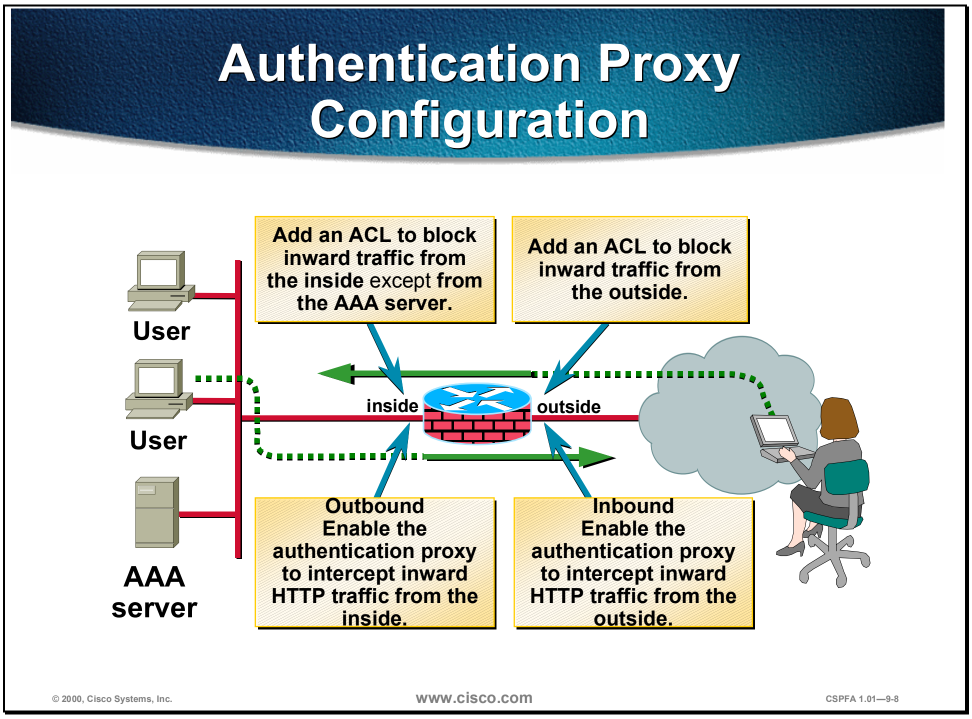
The authentication proxy sets up an inactivity (idle) timer for each user profile. As long as there is activity through the firewall, new traffic initiated from the user's host does not trigger the authentication proxy, and all authorized user traffic is permitted access through the firewall.

If the idle timer expires, the authentication proxy removes the user's profile information and dynamic access list entries. When this happens, traffic from the client host is blocked. The user must initiate another HTTP connection to trigger the authentication proxy.



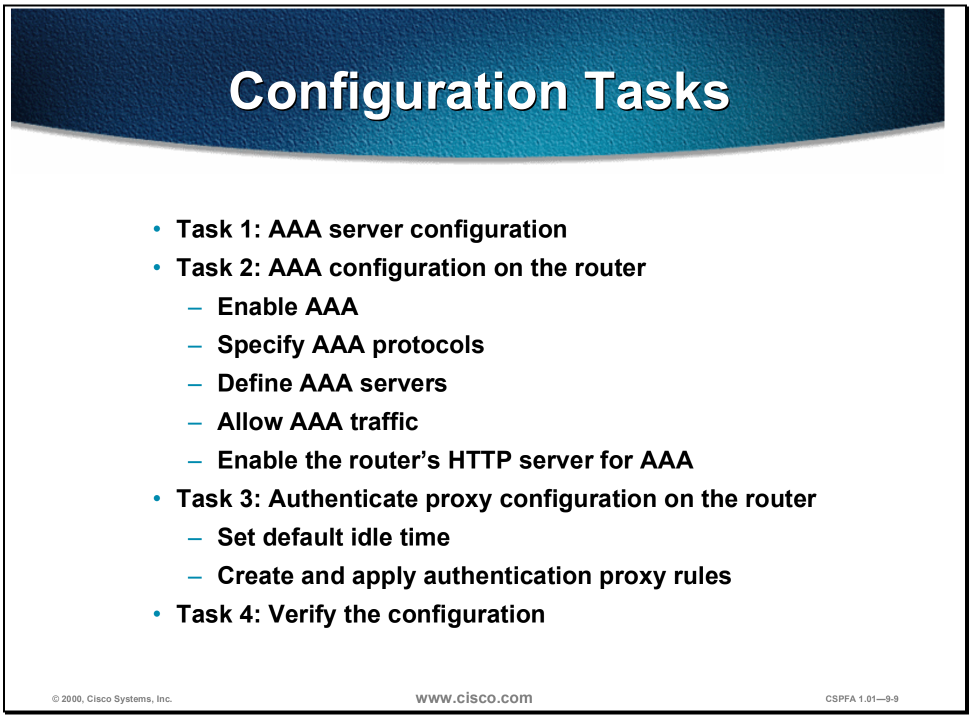
The Cisco IOS Firewall authentication proxy supports the following AAA protocols and servers:

- Terminal Access Controller Access Control System Plus (TACACS+)
  - Cisco Secure Asynchronous Communications Server (CSACS) for Windows NT (CSACS-NT)
  - Cisco Secure ACS for UNIX (CSACS-UNIX)
  - TACACS+ Freeware
- Remote Authentication Dial-In User Service (RADIUS)
  - Cisco Secure ACS for Windows NT (CSACS-NT)
  - Cisco Secure ACS for UNIX (CSACS-UNIX)
  - Livingston
  - Ascend



Apply the authentication proxy in the inward direction at any interface on the router where you want per-user authentication and authorization. Applying the authentication proxy inward at an interface causes it to intercept a user's initial connection request before that request is subjected to any other processing by the firewall. If the user fails to authenticate with the AAA server, the connection request is dropped.

How you apply the authentication proxy depends on your security policy. For example, you can block all traffic through an interface, and enable the authentication proxy feature to require authentication and authorization for all user-initiated HTTP connections. Users are authorized for services only after successful authentication with the AAA server. The authentication proxy feature also allows you to use standard access lists to specify a host or group of hosts whose initial HTTP traffic triggers the proxy.

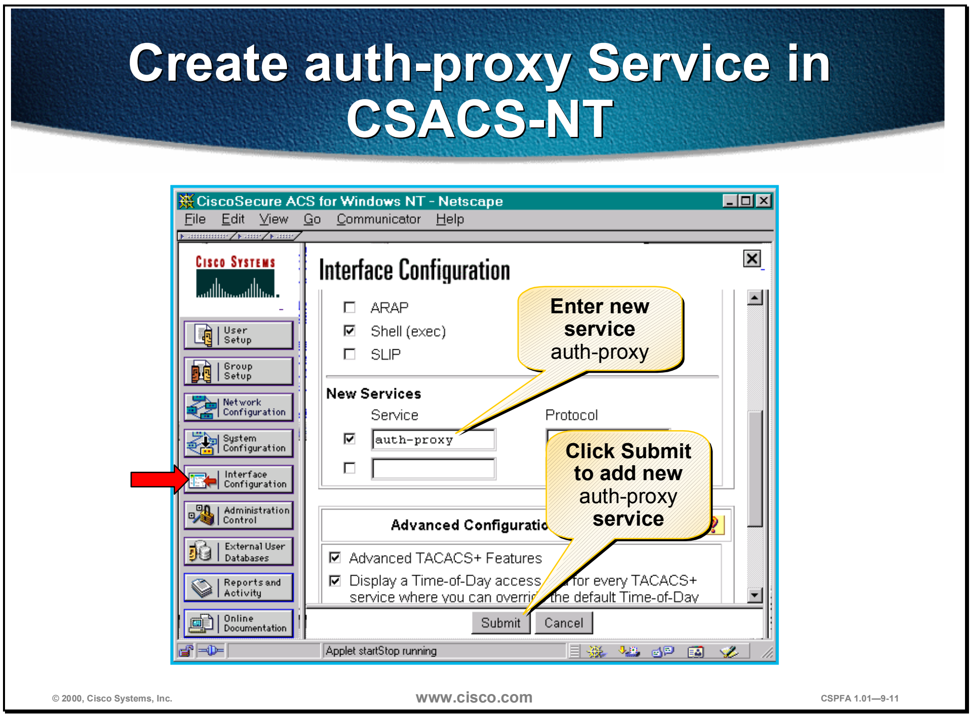


The following are the tasks to configure the authentication proxy:

- Task 1: AAA server configuration
- Task 2: AAA configuration on the router
  - Enable AAA
  - Specify AAA protocols
  - Define AAA servers
  - Allow AAA traffic
  - Enable the router’s HTTP server for AAA
- Task 3: Authenticate proxy configuration on the router
  - Set default idle time
  - Create and apply authentication proxy rules
- Task 4: Verify the configuration

AAA Server Configuration

This section discusses how to configure the AAA server to provide authentication and authorization for the Cisco IOS Firewall authorization proxy.



To support the authentication proxy, configure the AAA authorization service **auth-proxy** on the AAA server. This defines a separate section of authorization in the **Group Setup** section of the AAA for **auth-proxy** to specify the user profiles. This does not interfere with other type of services that the AAA server may have.

Complete the following steps to add authorization rules for specific services in Cisco Secure ACS:

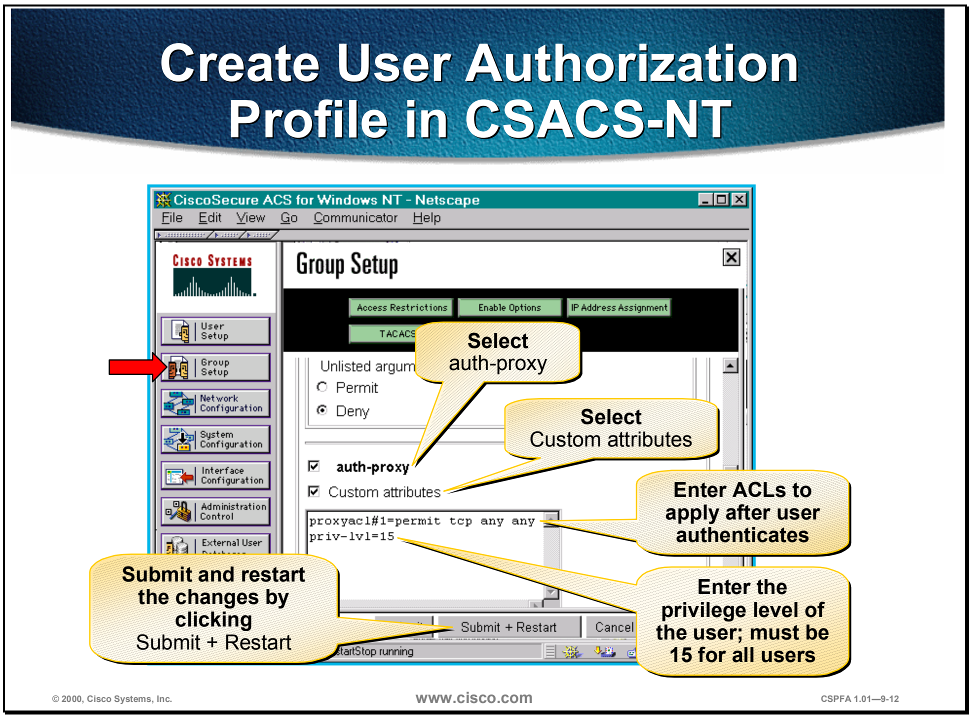
Step 1 In the navigation bar, click **Interface Configuration**. The Interface Configuration frame opens.

Step 2 Scroll down in the Interface Configuration frame until you find the New Services frame.

Step 3 Select the first checkbox in the Service column.

Step 4 Enter **auth-proxy** in the first empty Service field next to the checkbox you just selected.

Step 5 Click **Submit** when finished.



Step 6 In the navigation bar, click **Group Setup**. The ***Group Setup*** frame opens.

Step 7 Scroll down in the Group Setup frame until you find the newly created **auth-proxy** service.

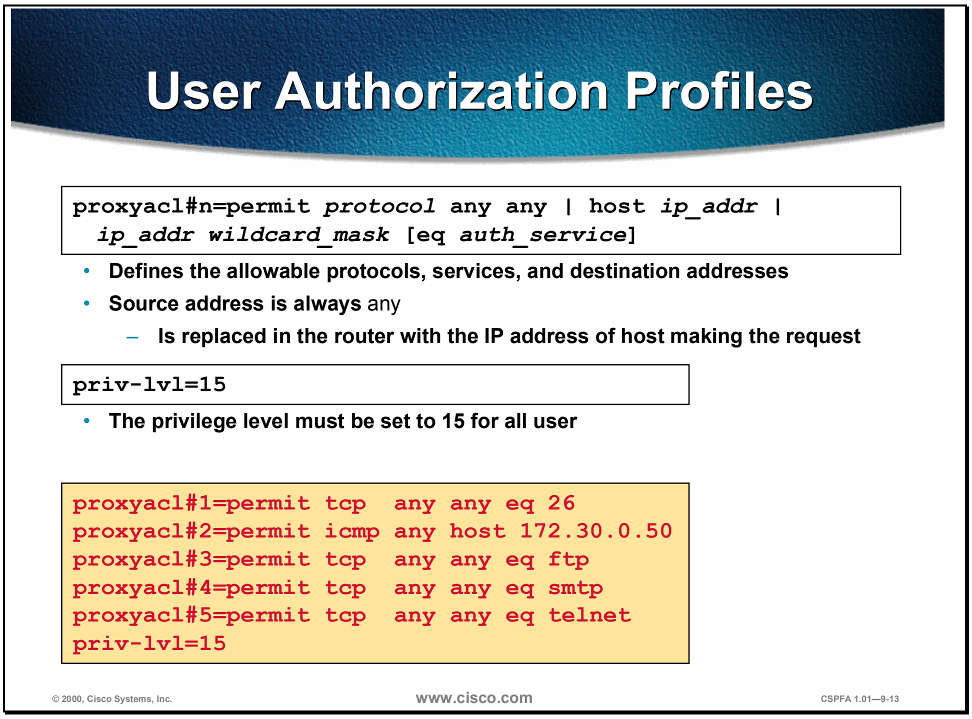
Step 8 Select the **auth-proxy** checkbox.

Step 9 Select the **Custom attributes** checkbox.

Step 10 Enter ACLs in the field below the Custom Attributes checkbox to apply after the user authenticates using the format from the following page.

Step 11 Enter the privilege level of the user (must be 15 for all users) using the format from the following page.

Step 12 Click **Submit + Restart** when finished.



Use the **proxyacl#n** attribute when configuring the access lists in the profile. The **proxyacl#n** attribute is for both RADIUS and TACACS+ attribute-value pairs. The access lists in the user profile on the AAA server must have **permit** access commands only. Set the source address to **any** in each of the user profile access list entries. The source address in the access lists is replaced with the source IP address of the host making the authentication proxy request when the user profile is downloaded to the firewall.

The following is the format for the ACLs used to enter in the Custom attributes box:

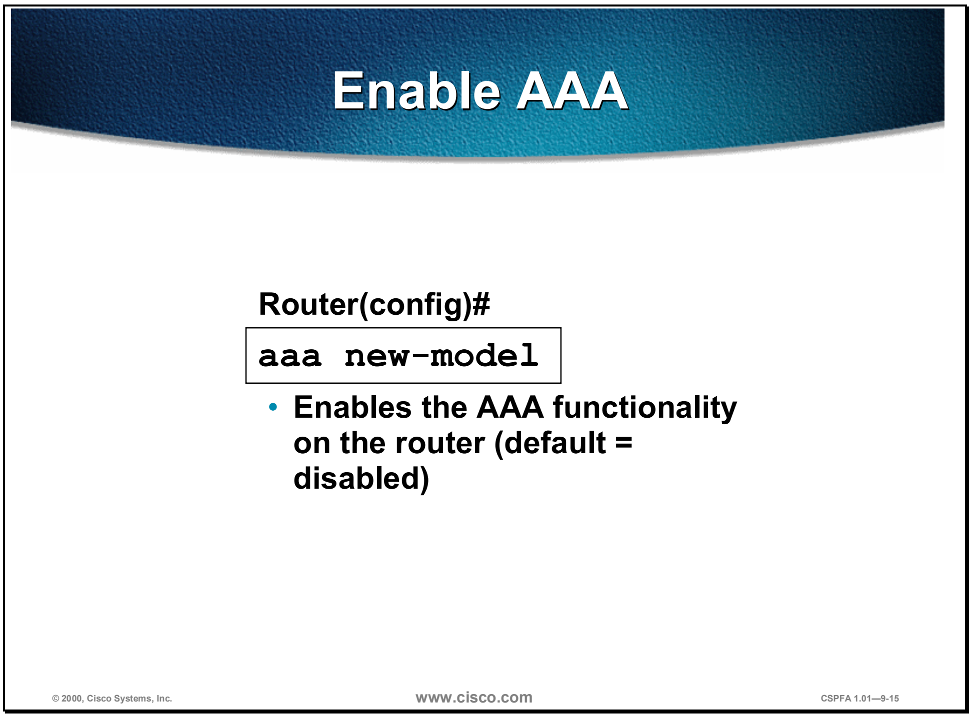
proxyacl#n=permit *protocol* any any | host *ip_addr* | *ip_addr wildcard_mask* [eq *auth_service*]

Arguments	Description
protocol	Keyword indicating the protocol to allow users to access. **tcp, udp, or icmp**.
any	Indicates any hosts. The first **any** after ***protocol*** is mandatory. This indicates any source IP address, which is actually replaced with the IP address of the user that requests authorization in the ACL applied in the router.
host *ip_addr*	IP address of a specific host that users can access.
ip_addr wildcard mask	IP address and wildcard mask for a network that users can access.
eq *auth_service*	Specific service that users are allowed to access.

Use **priv-lvl=15** to configure the privilege level of the authenticated user. The privilege level must be set to 15 for all users.

AAA Configuration

This section discusses how to configure The Cisco IOS Firewall to work with a AAA server and enable the authentication proxy feature.



Use the **aaa new-model** global configuration command to enable the AAA access control system. Use the **no** form of this command to disable the AAA access control model.

Note After you have enabled AAA, TACACS and extended TACACS commands are no longer available. If you initialize AAA functionality and later decide to use TACACS or extended TACACS, issue the **no** version of this command and then enable the version of TACACS that you want to use.

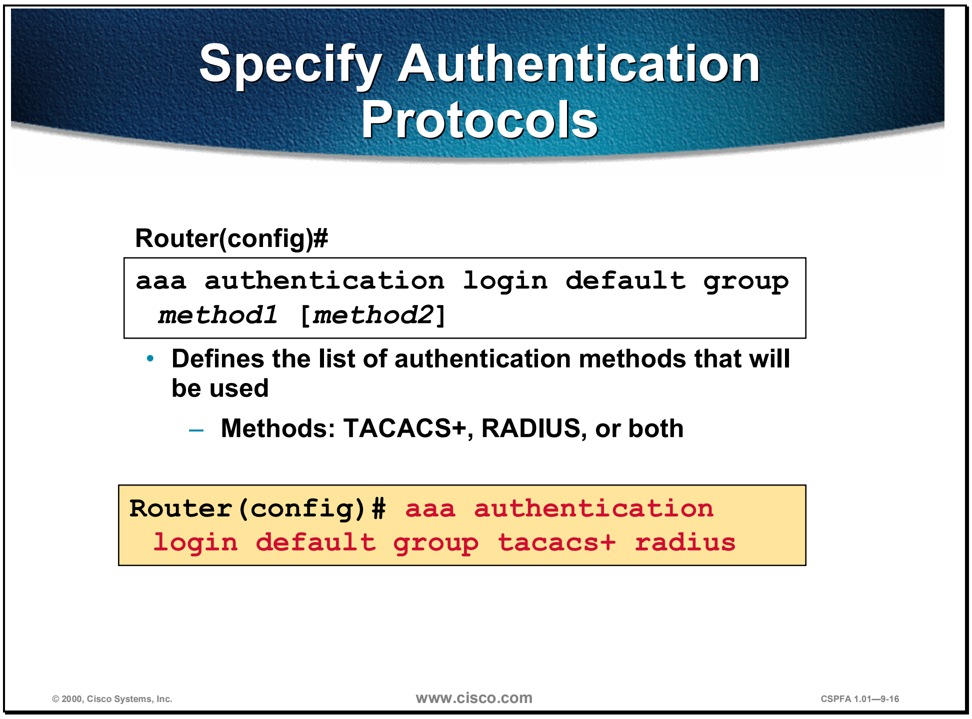
The syntax of the **aaa new-model** command is as follows:

aaa new-model

no aaa new-model

This command has no arguments.

By default, **aaa new-model** is not enabled.



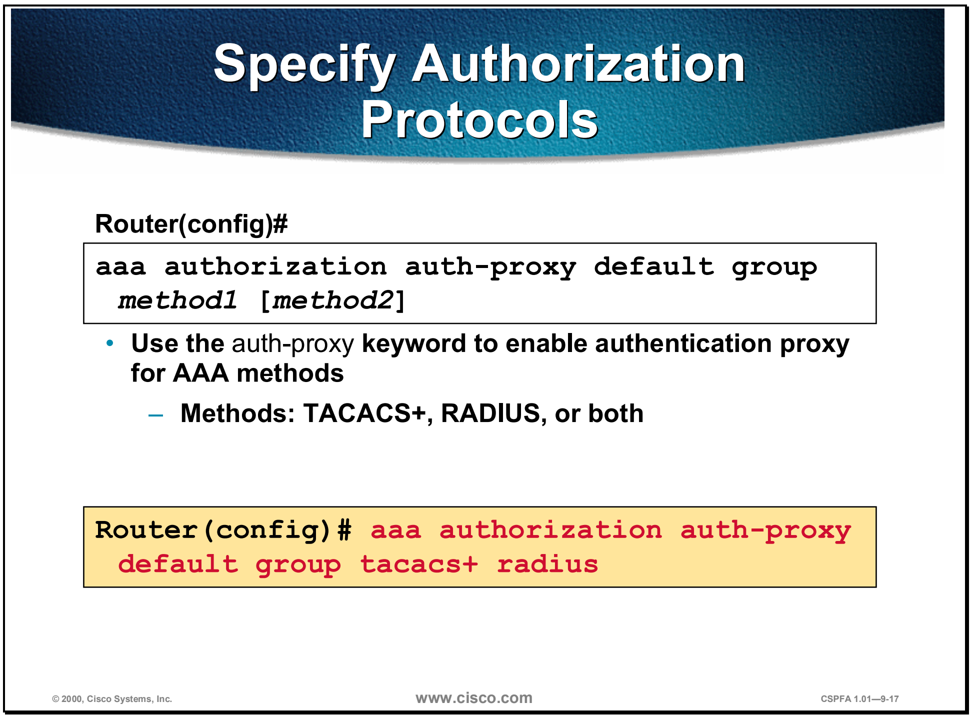
To set AAA authentication, use the **aaa authentication login** global configuration command. Use the **no** form of this command to disable AAA authentication.

The syntax of the **aaa authentication login** command is as follows:

aaa authentication login default group *method1* [*method2*]

no aaa authentication login default group *method1* [*method2*]

Argument	Description
method1, method2	The authentication protocols to use: **tacacs+**, **radius**, or both.



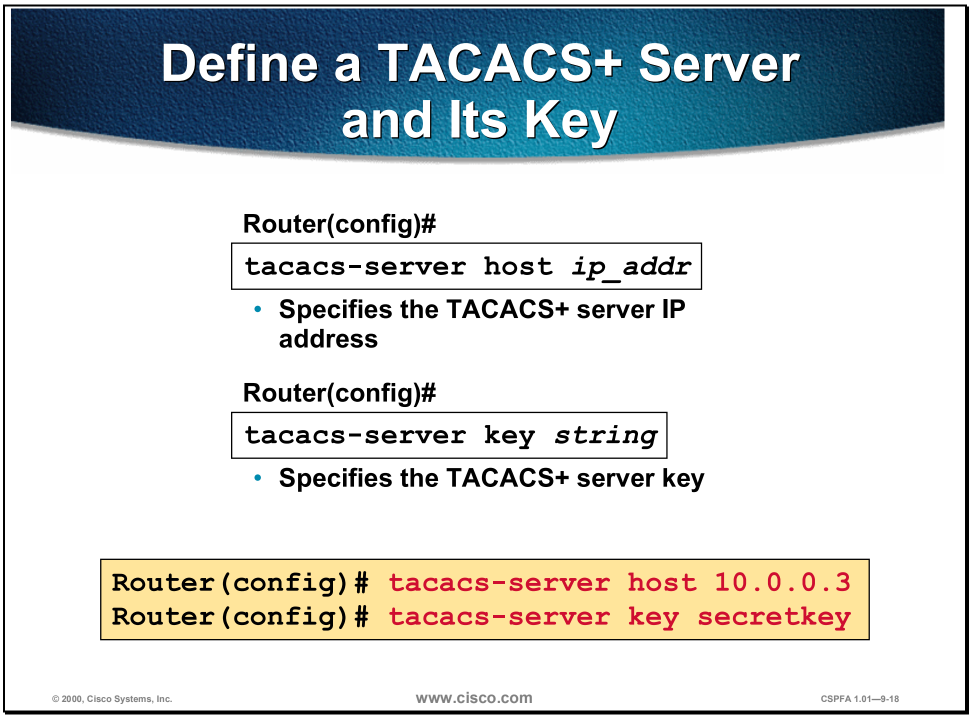
To set AAA authorization, use the **aaa authentication login** global configuration command. Use the **no** form of this command to disable AAA authentication.

The syntax of the **aaa authentication login** command is as follows:

aaa authorization auth-proxy default group ***method1*** **[*method2*]**

no aaa authorization auth-proxy default group ***method1*** **[*method2*]**

Argument	Description
method1, method2	The authorization protocols to use: **tacacs+**, **radius**, or both.



To specify the IP address of a TACACS+ server, use the **tacacs-server host** global configuration command. Use the **no** form of this command to delete the specified IP address. You can use multiple **tacacs-server host** commands to specify additional servers. The Cisco IOS Firewall software searches for servers in the order in which you specify them.

The syntax of the **tacacs-server host** command is as follows:

tacacs-server host ***ip_addr***

no tacacs-server host ***ip_addr***

Argument	Description
ip_addr	IP address of the TACACS+ server.

To set the authentication encryption key used for all TACACS+ communications between the Cisco IOS Firewall router and the AAA server, use the **tacacs-server key** global configuration command. Use the **no** form of this command to disable the key.

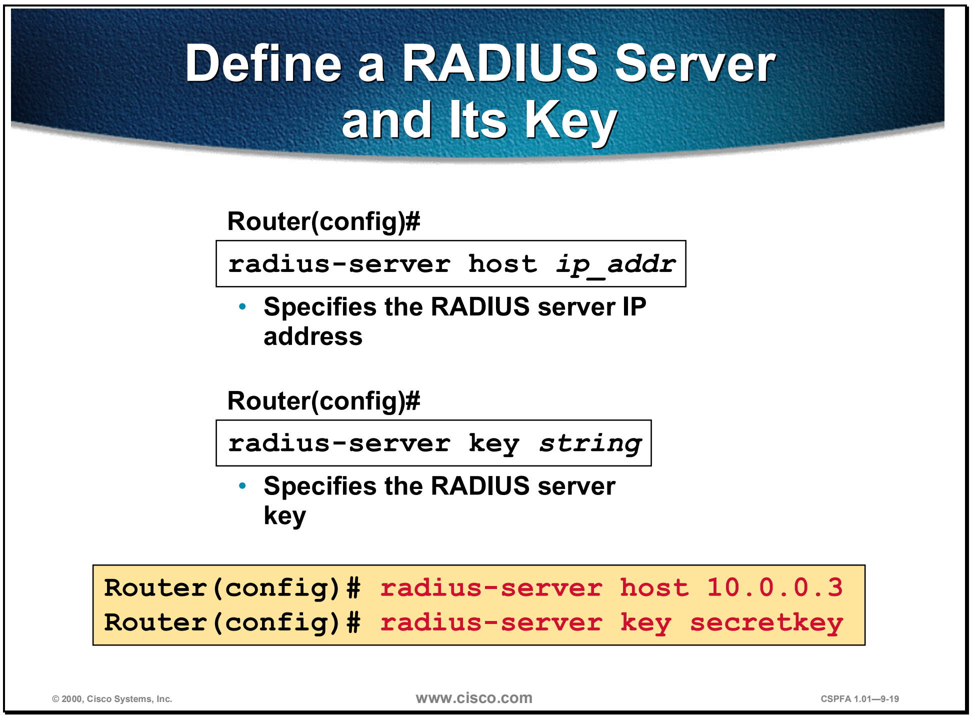
Note The key entered must match the key used on the AAA server. All leading spaces are ignored; spaces within and at the end of the key are not. If you use spaces in your key, do not enclose the key in quotation marks unless the quotation marks themselves are part of the key.

The syntax of the **tacacs-server key** command is as follows:

tacacs-server key ***string***

no tacacs-server key ***string***

Argument	Description
string	Key used for authentication and encryption.



To specify the IP address of a RADIUS server, use the **radius-server host** global configuration command. Use the **no** form of this command to delete the specified IP address. You can use multiple **radius-server host** commands to specify additional servers. The Cisco IOS Firewall software searches for servers in the order in which you specify them.

The syntax of the **radius-server host** command is as follows:

radius-server host ***ip_addr***

no radius-server host ***ip_addr***

Argument	Description
ip_addr	IP address of the RADIUS server.

To set the authentication encryption key used for all RADIUS communications between the Cisco IOS Firewall router and the AAA server, use the **radius-server key** global configuration command. Use the **no** form of this command to disable the key.

Note The key entered must match the key used on the AAA server. All leading spaces are ignored; spaces within and at the end of the key are not. If you use spaces in your key, do not enclose the key in quotation marks unless the quotation marks themselves are part of the key.

The syntax of the **radius-server key** command is as follows:

radius-server key ***string***

no radius-server key ***string***

Argument	Description
string	Key used for authentication and encryption.

Allow AAA Traffic to the Router

- Create an ACL to permit TACACS+ traffic from the AAA server to the firewall
  - Source address = AAA server
  - Destination address = interface where the AAA server resides
- May want to permit ICMP
- Deny all other traffic
- Apply the ACL to the interface on the side where the AAA server resides

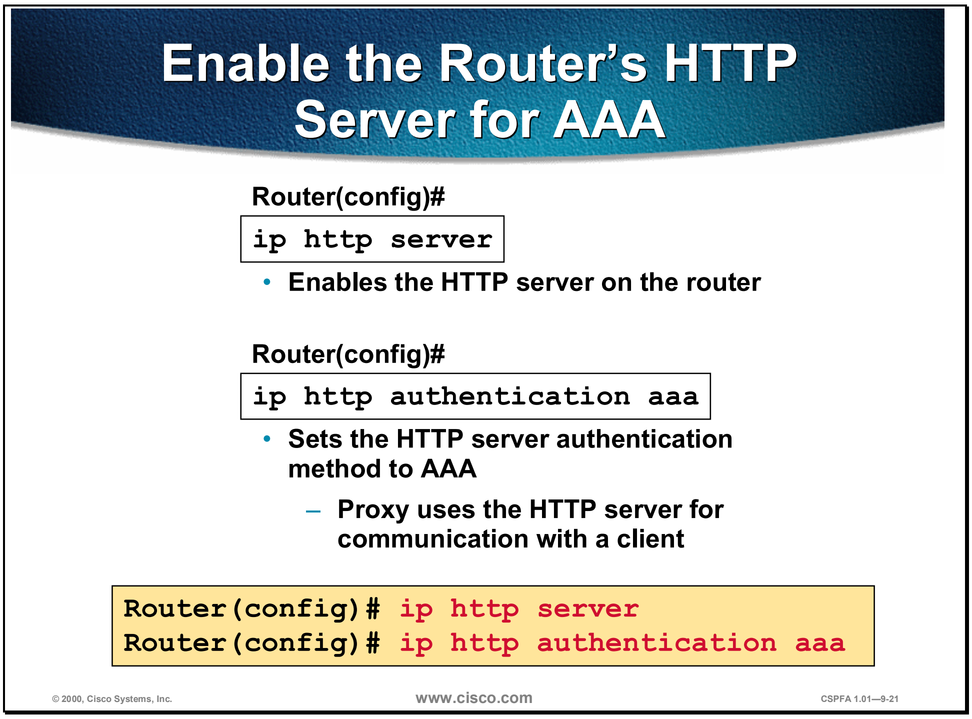
```
Router(config)# access-list 111 permit tcp 10.0.0.3 eq tacacs
  host 10.0.0.1
Router(config)# access-list 111 permit tcp 10.0.0.3 eq 1645
  host 10.0.0.1
Router(config)# access-list 111 permit icmp any any
Router(config)# access-list 111 deny ip any any
Router(config)# interface ethernet0/0
Router(config-if)# ip access-group 111 in
```

 www.cisco.com CSPFA 1.01—9-20

At this point you need to configure and apply an ACL to permit TACACS+ and RADIUS traffic from the AAA server to the firewall.

Use the following guidelines when writing the ACL:

- Source address = AAA server
- Destination address = interface where the AAA server resides
- May want to permit ICMP
- Deny all other traffic
- Apply the ACL to the interface on the side where the AAA server resides



To use the authentication proxy, use the **ip http server** command to enable the HTTP server on the router and the **ip http authentication aaa** command to make the HTTP server use AAA for authentication.

The syntax of the **ip http server** command is as follows:

ip http server

This command has no arguments.

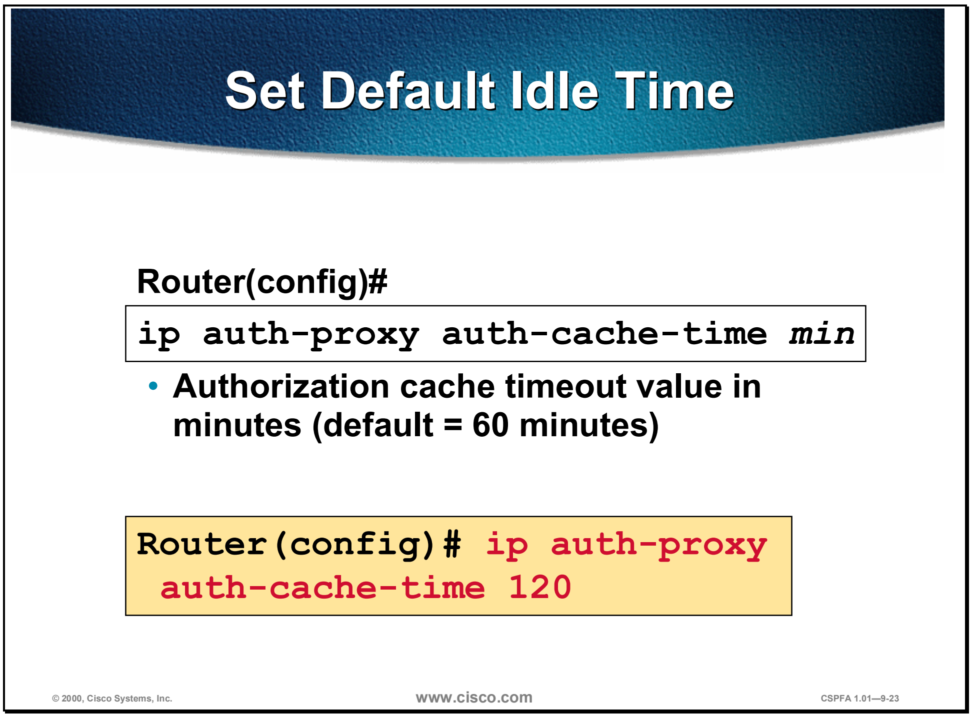
The syntax of the **ip http authentication aaa** command is as follows:

ip http authentication aaa

This command has no arguments.

Authentication Proxy Configuration

This section discusses how to configure the authentication proxy settings on a Cisco router.



To set the authentication proxy idle timeout value (the length of time an authentication cache entry, along with its associated dynamic user ACL, is managed after a period of inactivity), use the **ip auth-proxy auth-cache-time** global configuration command. To set the default value, use the **no** form of this command.

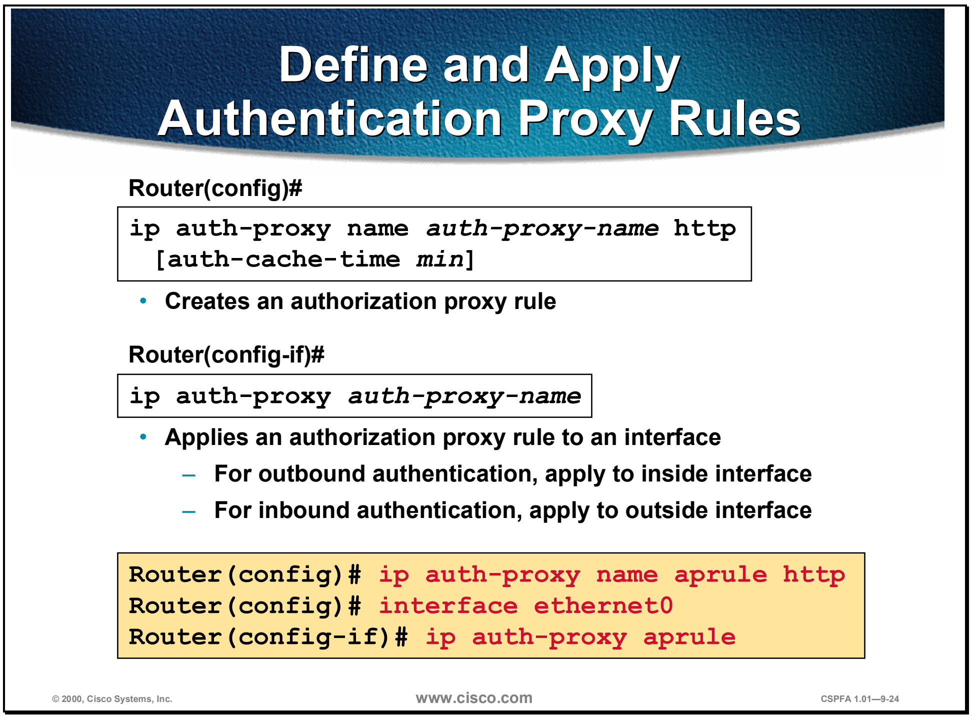
Note Set the **auth-cache-time** option for any authentication proxy rule to a higher value than the idle timeout value for any CBAC inspection rule. When the authentication proxy removes an authentication cache along with its associated dynamic user ACL, there might be some idle connections monitored by CBAC, and removal of user-specific ACLs could cause those idle connections to hang. If CBAC has a shorter idle timeout, CBAC resets these connections when the idle timeout expires; that is, before the authentication proxy removes the user profile.

The syntax of the **ip auth-proxy auth-cache-time** command is as follows:

ip auth-proxy auth-cache-time ***min***

no ip auth-proxy auth-cache-time

Argument	Description
min	Specifies the length of time, in minutes, that an authentication cache entry, along with its associated dynamic user ACL, is managed after a period of inactivity. Enter a value in the range of 1 to 2,147,483,647. The default value is 60 minutes.



To create an authentication proxy rule, use the **ip auth-proxy name** global configuration command. To remove the authentication proxy rules, use the **no** form of this command.

The syntax of the **ip auth-proxy name** command is as follows:

ip auth-proxy name ***auth-proxy-name*** **http [auth-cache-time** ***min*****]**

no ip auth-proxy name auth-proxy-name

Arguments	Description
auth-proxy-name	Associates a name with an authentication proxy rule. Enter a name of up to 16 alphanumeric characters.
auth-cache-time ***min***	(Optional) Overrides the global authentication proxy cache timer for a specific authentication proxy name, offering more control over timeout values. Enter a value in the range of 1 to 2,147,483,647. The default value is equal to the value set with the **ip auth-proxy auth-cache-time** command.

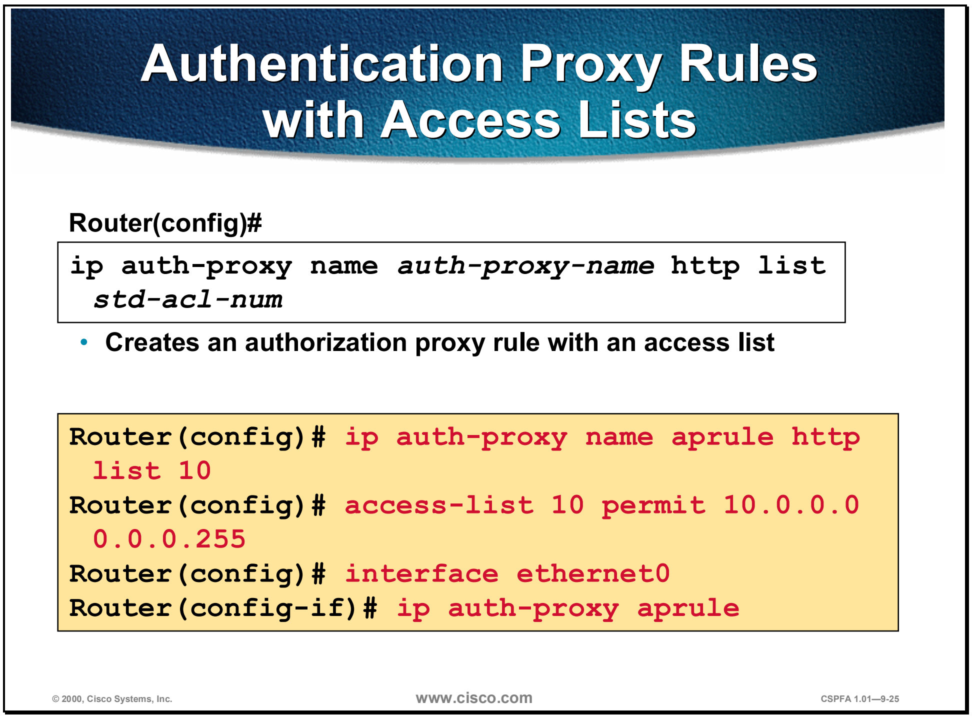
To apply an authentication proxy rule at a firewall interface, use the **ip auth-proxy** interface configuration command. To remove the authentication proxy rules, use the **no** form of this command.

The syntax of the **ip auth-proxy** command is as follows:

ip auth-proxy ***auth-proxy-name***

no ip auth-proxy auth-proxy-name

Arguments	Description
auth-proxy-name	Specifies the name of the authentication proxy rule to apply to the interface configuration. The authentication proxy rule is established with the **authentication proxy name** command.



You can associate an authentication proxy rule with an access control list, providing control over which hosts use the authentication proxy. To create an authentication proxy rule with ACLs, use the **ip auth-proxy name** global configuration command with the **list** ***std-acl-num*** option. To remove the authentication proxy rules, use the **no** form of this command.

The syntax of the **ip auth-proxy name** with ACLs command is as follows:

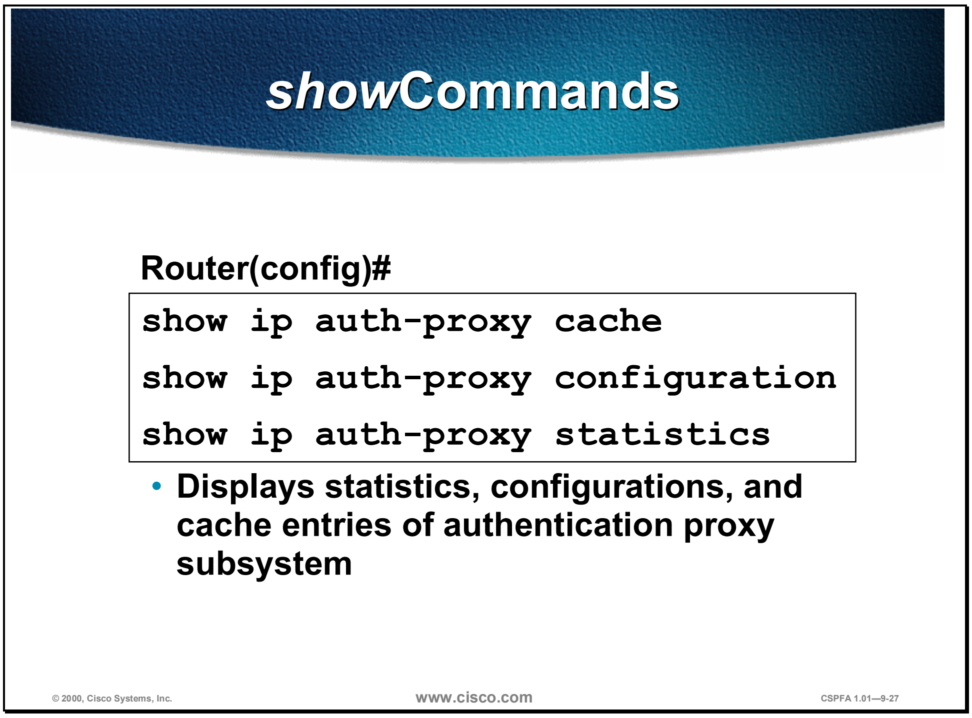
ip auth-proxy name ***auth-proxy-name*** **http list** ***std-acl-num***

no ip auth-proxy name auth-proxy-name

Arguments	Description
auth-proxy-name	Associates a name with an authentication proxy rule. Enter a name of up to 16 alphanumeric characters.
list *std-acl-num*	Specifies a standard access list to use with the authentication proxy. With this option, the authentication proxy is applied only to those hosts in the standard access list. If no list is specified, all connections initiating HTTP traffic arriving at the interface are subject to authentication.

Test and Verify the Configuration

This section discusses the procedures for testing and verifying the authentication proxy configuration.

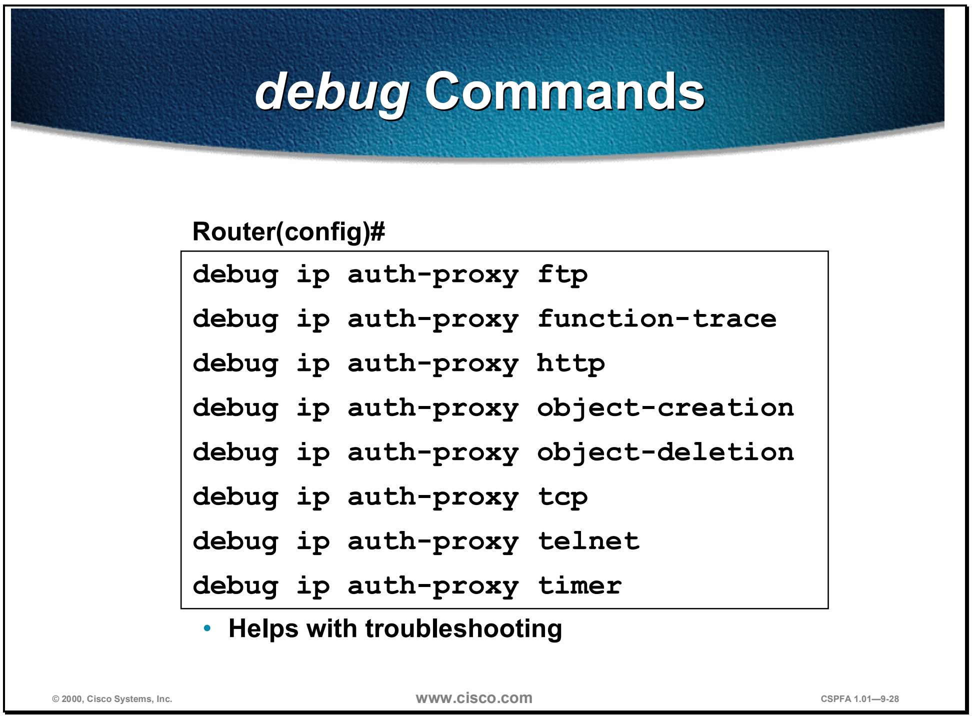


Use the **show ip auth-proxy** command to display the authentication proxy entries, the running authentication proxy configuration, or the authentication proxy statistics.

The syntax of the **show ip auth-proxy** command is as follows:

show ip auth-proxy cache | **configuration** | **statistics**

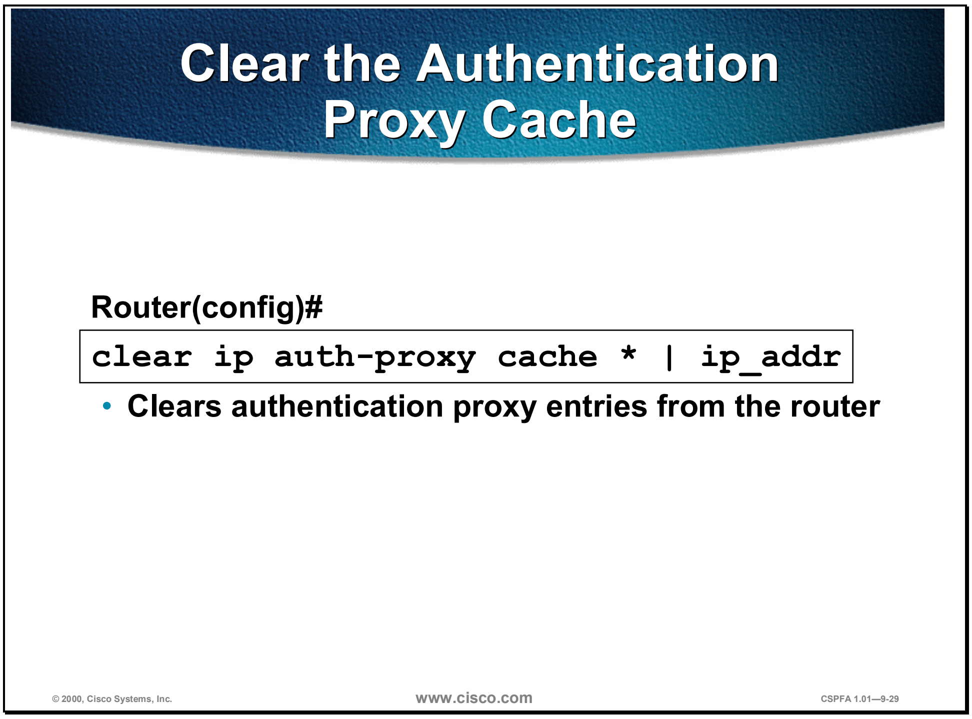
Arguments	Description
cache	Lists the host IP address, the source port number, the timeout value for the authentication proxy, and the state for connections using authentication proxy. If the authentication proxy state is HTTP_ESTAB, the user authentication was successful.
configuration	Displays all authentication proxy rules configured on the router.
statistics	Displays all the router statistics related to the authentication proxy.



The syntax of the **debug ip auth-proxy** command is as follows:

debug ip auth-proxy ftp | **function-trace** | **http** | **object-creation** | **object-deletion** | **tcp** | **telnet** | **timer**

Arguments	Description
ftp	Displays FTP events related to the authentication proxy.
function-trace	Displays the authentication proxy functions.
http	Displays HTTP events related to the authentication proxy.
object-creation	Displays additional entries to the authentication proxy cache.
object-deletion	Displays deletion of cache entries for the authentication proxy.
tcp	Displays TCP events related to the authentication proxy.
telnet	Displays Telnet-related authentication proxy events.
timer	Displays authentication proxy timer-related events.



The syntax of the **clear ip auth-proxy cache** command is as follows:

clear ip auth-proxy cache * | ***ip_addr***

Argument	Description
*	Clears all authentication proxy entries, including user profiles and dynamic access lists.
ip_addr	Clears the authentication proxy entry, including user profiles and dynamic access lists, for the specified IP address.

Lab Exercise: Configure Authentication Proxy on a Cisco Router

Complete the following lab exercise to practice what you have learned in this chapter.

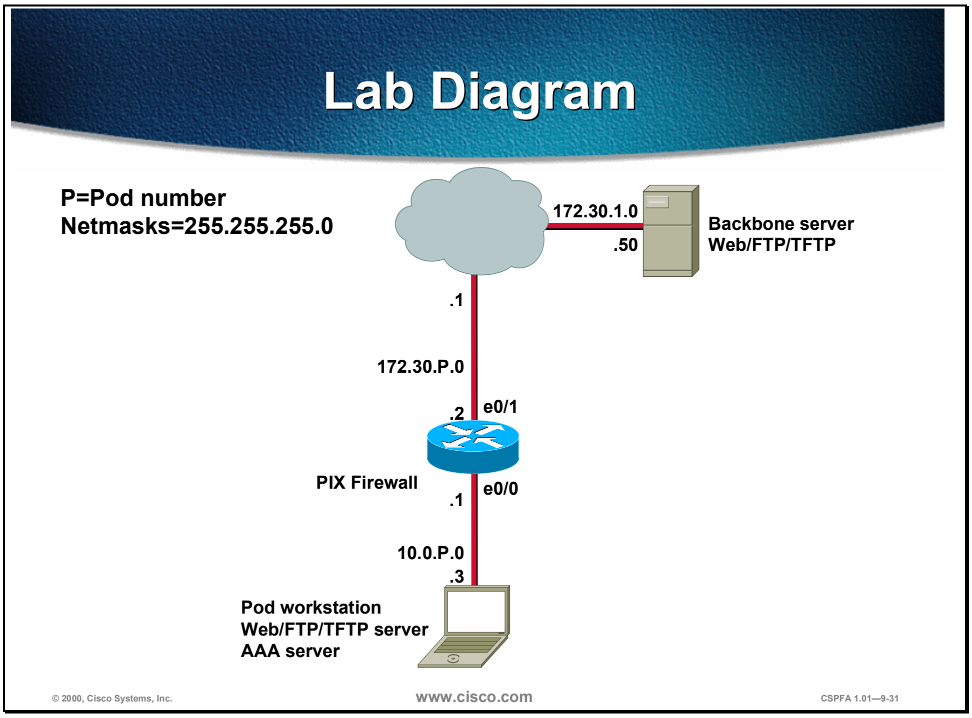
Objectives

In this lab you will complete the following tasks:

- Configure Cisco Secure ACS NT
- Configure AAA
- Configure authentication proxy
- Test and verify authentication proxy

Lab Diagram

The following figure displays the pod configuration that you will use to complete in this lab exercise.



Task 1: Configure Cisco Secure ACS NT

Step 1 On your workstation, open Cisco Secure ACS from the desktop.

Step 2 Click **Interface Configuration** on the left-hand column of CSACS to go to the Interface Configuration window.

Step 3 Click **TACACS+ (Cisco)** to configure this option.

Step 4 Scroll down until you find New Services.

Step 5 Select the first line under New Service and enter **`auth-proxy`** under service.

Step 6 Choose **"Advanced TACACS+ features"**.

Step 7 Click **Submit** to submit your changes.

Step 8 Click **Group Setup** to open the Group Setup window.

Step 9 Select **Default Group (1 user)** under the Group pull-down menu.

Step 10 Click **Edit Settings** to go to the "Group Setup" for this group.

Step 11 Scroll down until you find the auth-proxy checkbox followed by the Custom attributes checkbox. Check both the auth-proxy checkbox and the Custom attributes checkbox.

Step 12 Enter the following in the Custom attributes box:

```
proxyacl#1=permit tcp any any
priv-lvl=15
```

Step 13 Click **Submit + Restart** to submit your changes and restart CSACS. Wait for the interface to return to the Group Setup main window.

Task 2: Configure AAA

Step 1 On your router, enter global configuration mode:

```
Router# configure terminal
```

Step 2 Enable AAA:

```
Router(config)# aaa new-model
```

Step 3 Specify the authentication protocol:

```
Router(config)# aaa authentication login default group tacacs+
```

Step 4 Specify the authorization protocol:

```
Router(config)# aaa authorization auth-proxy default group tacacs+
```

Step 5 Define the TACACS+ server and its key:

```
Router(config)# tacacs-server host 10.0.P.3 (P=pod num.)
Router(config)# tacacs-server key secretkey
```

Step 6 Clear the previously applied access list on the inside interface:

```
Router(config)# no access-list 101
```


Step 7 Define a new access list to allow TACACS+ traffic to the inside interface from your AAA server. Also allow outbound ICMP traffic and CBAC traffic (FTP and WWW). Block all other inside-initiated traffic:

```
Router(config)# access-list 101 permit tcp host 10.0.P.3 eq tacacs host 10.0.P.1
Router(config)# access-list 101 permit icmp any any
Router(config)# access-list 101 permit tcp 10.0.P.0 0.0.0.255 any eq ftp
Router(config)# access-list 101 permit tcp 10.0.P.0 0.0.0.255 any eq www
Router(config)# access-list 101 deny ip any any
```

(where P = pod number, and Q = peer pod number)

Step 8 Enable the router's HTTP server for AAA:

```
Router(config)# ip http server
Router(config)# ip http authentication aaa
```

Task 3: Configure Authentication Proxy

Step 1 Define an authentication proxy rule:

```
Router(config)# ip auth-proxy name APRULE http auth-cache-time 5
```

Step 2 Apply the authentication proxy rule to the inside interface:

```
Router(config)# interface ethernet 0/0
Router(config-if)# ip auth-proxy APRULE
Router(config-if)# end
```

Task 4: Test and Verify Authentication Proxy

Step 1 On your router, use the **show access-list** command to check your access lists. Fill in the blanks below using the output from this command:

```
Router# show access-list
Extended IP access list 101
_________________________________________________________________
_________________________________________________________________
_________________________________________________________________

Extended IP access list 102
_________________________________________________________________
_________________________________________________________________
_________________________________________________________________
_________________________________________________________________
```

Step 2 On your router, use the **show ip inspect** command to see CBAC sessions. Fill in the blanks below using the output from this command:

```
Router# show ip inspect sessions
_________________________________________________________________
_________________________________________________________________
```

Step 3 Use the **show ip auth-proxy configuration** command to verify the authorization proxy configuration. Fill in the blanks below using the output from this command:

```
Router# show ip auth-proxy configuration
Authentication global cache time is _____ minutes
Authentication Proxy Rule Configuration
```

```
Auth-proxy name ___________
   http list not specified auth-cache-time _____ minutes
```

Step 4 Use the **show ip auth-proxy statistics** command to verify the authorization proxy statistics. Fill in the blanks below using the output from this command:

```
Router# show ip auth-proxy statistics
Authentication Proxy Statistics
    proxied client number _____
```

Step 5 Use the **show ip auth-proxy cache** command to verify the authorization proxy configuration. Fill in the blanks below using the output from this command:

```
Router# show ip auth-proxy cache
_________________________________________________________
```

Step 6 From your workstation command prompt, ping the backbone server:

```
C:\> ping 172.30.1.50
Pinging 172.30.1.50 with 32 bytes of data:

Reply from 172.30.1.50: bytes=32 time=34ms TTL=125
Reply from 172.30.1.50: bytes=32 time=34ms TTL=125
Reply from 172.30.1.50: bytes=32 time=34ms TTL=125
Reply from 172.30.1.50: bytes=32 time=36ms TTL=125
```

Step 7 Use your Web browser to connect to the backbone Web server. In the URL field enter:

http://172.30.1.50

Step 8 Enter the following when the Web browser prompts you for a username and password:

```
Username: aaauser
Password: aaapass
```

Step 9 Use the **show access-list** command to check your access lists. Fill in the blanks below using the output from this command:

```
Router# show access-list
Extended IP access list 101
_________________________________________________________
_________________________________________________________
_________________________________________________________
Extended IP access list 102
_________________________________________________________
_________________________________________________________
_________________________________________________________
_________________________________________________________
_________________________________________________________
_________________________________________________________
_________________________________________________________
_________________________________________________________
```

On your router, use the show ip inspect sessions **command to see CBAC sessions:**

```
Router# show ip inspect sessions
_________________________________________________________
_________________________________________________________
```


____
____
____

Step 10 Use the **show ip auth-proxy statistics** command to verify the authorization proxy statistics. Fill in the blanks below using the output from this command:

```
Router# show ip auth-proxy statistics
Authentication Proxy Statistics
    proxied client number _____
```

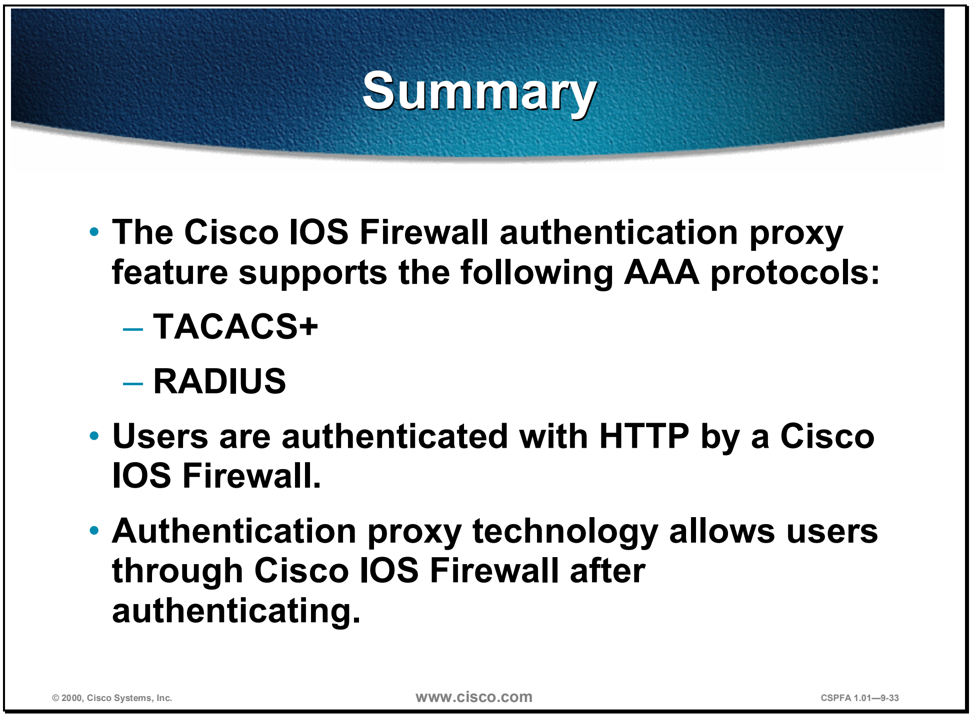
Step 11 Use the **show ip auth-proxy cache** command to verify the authorization proxy configuration. Fill in the blanks below using the output from this command:

```
Router# show ip auth-proxy cache
```

____
____

Summary

This section summarizes what you have learned in this chapter.



A

Image Upgrade and Password Recovery for the PIX Firewall 515 and 520

Image Upgrade

This section explains how to upgrade your PIX Firewall image.

PIX Firewall 515 Image Upgrade

Complete the following steps to upgrade the PIX Firewall image:

Step 1 Interrupt the boot process to enter monitor mode by pressing the **Escape** key or sending a **Break** character.

Step 2 Specify the PIX Firewall interface to use for tftp. To do this, you must enter the following command at the monitor prompt:

```
monitor> interface [num]
```

Step 3 Specify the IP address of the PIX Firewall:

```
monitor> address [IP_address]
```

Step 4 Specify default gateway (if needed):

```
monitor> gateway [IP_address]
```

Step 5 Verify connectivity to the TFTP server:

```
monitor> ping [server_addres]
```

Step 6 Name the server:

```
monitor> server [IP_address]
```

Step 7 Name the image filename:

```
monitor> file [name]
```

Step 8 Start the TFTP process:

```
monitor> tftp
```

PIX Firewall 520 Image Upgrade

Complete the following steps for the image upgrade of the PIX Firewall 520 to versions lower than 5.1:

Step 1 Download the file for the PIX Firewall software version you are running from Cisco Connection Online (CCO) (each version requires a different file): ftp://ftp.cisco.com /cisco/internet/pix/special/your version.

You will need a CCO login to download this data.

Step 2 Download the rawrite.exe file into the same directory as the password version you downloaded previously.

Step 3 Execute the rawrite.exe file as follows and enter the information when prompted:

```
C:\> rawrite
RaWrite 1.2 – Write disk file to a floppy diskette
```

```
Enter the source file name: pixXXX.bin (where XXX=version number)
Enter the destination drive: a:
Please insert a formatted diskette into drive A: and press-ENTER- : <Enter>
Number of sectors per track for this disk is 18
Writing image to drive A:. Press ^C to abort.
Track: 78 Head: 1 Sector: 16
Done.
C:\>
```

Step 4 Reboot your PIX Firewall with the diskette you just created. The system automatically loads the new image into Flash memory. Remove the diskette after rebooting. You are finished with the upgrade.

Complete the following steps for the image upgrade of the PIX Firewall 520 to versions 5.1 and higher:

Step 1 Download the file for the PIX Firewall software version you are running from CCO (each version requires a different file):

ftp://ftp.cisco.com /cisco/internet/pix/special/your version.

You will need a CCO login to download this data.

Step 2 Download the boothelper utility file for the software version you are upgrading to from CCO (each version requires a different file):

ftp://ftp.cisco.com /cisco/internet/pix/special/your version.

Step 3 Download the rawrite.exe file into the same directory as the password version you downloaded previously.

Step 4 Execute the rawrite.exe file as follows and enter the information when prompted:

```
C:\> rawrite
RaWrite 1.2 – Write disk file to a floppy diskette

Enter the source file name: bhXXX.bin (where XXX=version number)
Enter the destination drive: a:
Please insert a formatted diskette into drive A: and press-ENTER- : <Enter>
Number of sectors per track for this disk is 18
Writing image to drive A:. Press ^C to abort.
Track: 78 Head: 1 Sector: 16
Done.
C:\>
```

Step 5 Reboot your PIX Firewall with the diskette you just created. The system automatically loads the boothelper utility from which you will TFTP the new image to Flash memory.

Step 6 At the boothelper prompt, specify the PIX Firewall interface to use for TFTP:

```
boothelper> interface [num]
```

Step 7 Specify the IP address of the PIX Firewall interface's IP address:

```
boothelper> address [IP_address]
```

Step 8 Specify the default gateway (if needed):

```
boothelper> gateway [IP_address]
```

Step 9 Verify connectivity to the TFTP server:

```
boothelper> ping [server_addres]
```

Step 10 Name the server:

```
boothelper> server [IP_address]
```

Step 11 Name the image filename:

```
boothelper> file [name]
```

Step 12 Start the TFTP process.:

```
boothelper> tftp
```

Step 13 The system automatically reboots with the new image in Flash memory. Quickly remove the boothelper diskette when prompted to do so. You are finished with the upgrade.

Password Recovery

This section explains how to recover your password for the PIX Firewall 515 and 520.

PIX Firewall 520 Password Recovery

The password recovery for the PIX Firewall 520 requires writing of a special image to a floppy diskette. Use this diskette to boot PIX Firewall 520. Complete the following steps to perform a PIX Firewall 520 password recovery:

Step 1 Download the file for the PIX Firewall software version you are running from CCO (each version requires a different file):

ftp://ftp.cisco.com /cisco/internet/pix/special/your version.

You will need a CCO login to download this data.

Step 2 Download the rawrite.exe file into the same directory as the password version you downloaded previously.

Step 3 After you have retrieved the two files, execute the rawrite.exe file as follows and enter the information when prompted:

```
C:\> rawrite
RaWrite 1.2 – Write disk file to a floppy diskette

Enter the source file name: npXXX.bin (where XXX=version number)
Enter the destination drive: a:
Please insert a formatted diskette into drive A: and press -ENTER- : <Enter>
Number of sectors per track for this disk is 18
Writing image to drive A:. Press ^C to abort.
Track: 78 Head: 1 Sector: 16
Done.
C:\>
```

Step 4 Reboot your PIX Firewall with the diskette you just created. When prompted, press **y** to erase the password.

```
Do you wish to erase the passwords? [yn] y
Passwords have been erased
```

The system automatically erases the password and starts rebooting.

PIX Firewall 515 Password Recovery

The password recovery for the PIX Firewall 515 requires a TFTP sever. Complete the following steps to perform a PIX Firewall 520 password recovery:

Step 1 Download the file for the PIX Firewall software version you are running from CCO (each version requires a different file):

ftp://ftp.cisco.com /cisco/internet/pix/special/your version.

You will need a CCO login to download this data.

Step 2 Move the binary file you just downloaded to the TFTP home folder in your TFTP server.

Step 3 Reboot your PIX Firewall and interrupt the boot process to enter monitor mode by pressing the **Escape** key or sending a **Break** character.

Step 4 Specify the PIX Firewall interface to use for TFTP:

```
monitor> interface [num]
```

Step 5 Specify the IP address of the PIX Firewall interface's IP address:

```
monitor> address [IP_address]
```

Step 6 Specify the default gateway (if needed:

```
monitor> gateway [IP_address]
```

Step 7 Verify connectivity to the TFTP:

```
monitor> ping [server_addres]
```

Step 8 Name the server:

```
monitor> server [IP_address]
```

Step 9 Name the image filename:

```
monitor> file [name]
```

Step 10 Start the TFTP process:

```
monitor> tftp
```

Step 11 When prompted, press **y** to erase the password.

```
Do you wish to erase the passwords? [yn] y
Passwords have been erased
```

The system automatically erases the password and starts rebooting.

