



Managing Network Access and Use

This chapter describes how to establish and control network connectivity for different applications and implementations after you have completed your basic configuration, described in [Chapter 2, “Basic Firewall Configuration.”](#) This chapter contains the following sections:

- [Assigning a Fixed Address to a Server](#)
- [Allowing Inbound Connections](#)
- [Controlling Outbound Connections](#)
- [Using Authentication and Authorization](#)
- [Filtering Outbound Connections](#)
- [Advanced Configuration Example](#)

Assigning a Fixed Address to a Server

Static address translation creates a permanent, one-to-one mapping between an address on an internal network (a higher security level interface) and a perimeter or external network (lower security level interface). For example, to share a web server on a perimeter interface with users on the public Internet, use static address translation to map the server’s actual address to a registered IP address. Static address translation hides the actual address of the server from users on the less secure interface, making casual access by unauthorized users less likely. Unlike NAT or PAT, it requires a dedicated address on the outside network for each host, so it does not save registered IP addresses.

If you use a **static** command to allow inbound connections to a fixed IP address, use the **access-list** and **access-group** commands to create an access list and to bind it to the appropriate interface. For more information, refer to [“Allowing Inbound Connections.”](#)



Note

The **static** command may not be compatible with failover. When creating a static mapping to an interface, the failover in that interface will be in the waiting state because it does not receive a response from the other PIX Firewall unit for the failover-specific protocol 105. For further information about failover, refer to [Chapter 8, “Using PIX Firewall Failover.”](#)

The main options of the **static** command are as follows:

```
static [(internal_if_name, external_if_name)] global_ip local_ip [netmask network_mask]  
[max_conns]
```

- Replace *internal_if_name* with the internal network interface name. The higher security level interface you are accessing.
- Replace *external_if_name* with the external network interface name. The lower security level interface you are accessing.
- Replace *global_ip* with the outside (global) IP address. This is the interface with the lower security level. This address cannot be a PAT IP address.
- Replace *local_ip* with the internal (local) IP address from the inside network. This is the interface with the higher security level.
- Replace *network_mask* with the network mask pertains to both *global_ip* and *local_ip*. For host addresses, always use 255.255.255.255. For network addresses, use the appropriate subnet mask for the network.
- (Optional) replace *max_conns* with the maximum number of concurrent connections permitted through the static address translation.

For example, the following command maps a server with an internal IP address of 10.1.1.3 to the registered IP address 209.165.201.12:

```
static (inside, outside) 209.165.201.12 10.1.1.3 netmask 255.255.255.255 0 0
```

This command simply maps the addresses; make sure you also configure access using the **access-list** and **access-group** commands, as described in the next section. Also, you will need to inform the DNS administrator to create an MX record for the external address so that traffic sent to the server host name is directed to the correct address.

**Note**

For information about how to configure static translation without NAT, refer to the static command page in the *Cisco PIX Firewall Command Reference*.

Allowing Inbound Connections

By default, the PIX Firewall denies access to an internal or perimeter (more secure) network from an external (less secure) network. You specifically allow inbound connections by using access lists. Access lists work on a first-match basis, so for inbound access, you need to deny first and then permit after.

**Note**

Beginning with PIX Firewall version 5.3, access lists are the preferred method for managing network access. The **conduit** command was used in earlier versions. Access lists provide improved flexibility and greater ease of use for those familiar with Cisco IOS access control. However, the **conduit** command is still supported to maintain backward compatibility of configurations written for previous PIX Firewall versions.

You use the **access-list** and **access-group** commands to permit access based on source or destination IP address, or by the protocol port number. Use the **access-list** command to create a single access list entry, and use the **access-group** command to bind one or more access list entries to a specific interface. Only specify one **access-group** command for each interface.

**Note**

To allow access only for specific users, set up authentication, as described in [“Using Authentication and Authorization.”](#)

Before you can set up an access list for a host, set up address translation by using a **global** or **static** command. Setting up address translation with the **global** command is described in [Chapter 2, “Basic Firewall Configuration.”](#) Setting up address translation using the static command was described earlier in the previous section “[Assigning a Fixed Address to a Server.](#)”

The format for the access-list command is as follows:

```
access-list ID action protocol source_address port destination_address port
```

- Replace *ID* with a name or number you create to identify a group of **access-list** command statements; for example, “acl_out,” which identifies that the permissions apply to access from the outside interface.
- Replace *action* with **permit** or **deny** depending on whether you want to permit or deny access to the server. By default, all inbound access is denied, so you will need to permit access to a specific protocol or port.
- Replace *protocol* with the protocol (tcp or udp). For most servers, such as HTTP or email, use **tcp**.
- Replace *source_address* with the host or network address for those systems on the lower security level interface that need to access the *destination_address*. Use **any** to let any host access the *destination_address*. If you specify a single host, precede the address with **host**; for example **host 192.168.1.2**. If you specify a network address, also specify a network mask; for example, **192.168.1.0 255.255.255.0**.
- Replace the first *port* parameter with the protocol port used by the source host to initiate the connection.
- Replace *destination_address* with the host or network global address that you specified with the **static** command statement. For a host address, precede the address with **host**; for networks, specify the network address and the appropriate network mask.
- Replace the second *port* parameter with the literal port name or number for the destination server protocol. For a web server, use **the string http** or the number 80. For an email server, use **smtp** or the number 25. The port is preceded with the **eq** (equals) parameter.

The following is a list of literal port names that you can use when configuring an **access-list** command statement: DNS, ESP, FTP, H323, HTTP, IDENT, NNTP, NTP, POP2, POP3, PPTP, RPC, SMTP, SNMP, SNMPTRAP, SQLNET, TCP, Telnet, TFTP, and UDP. You can also specify these ports by number. Port numbers are defined in RFC 1700.

Two **access-list** command statement definitions are required to permit access to the following ports:

- DNS, Discard, Echo, Ident, NTP, RPC, SUNRPC, and Talk each require one definition for TCP and one for UDP.
- PPTP requires one definition for port 1723 on TCP and another for port 0 and GRE.
- TACACS+ requires one definition for port 65 on TCP and another for port 49 on UDP.

The format for the **access-group** command is as follows:

```
access-group ID in interface low_interface
```

Replace *ID* with the same identifier that you specified in the **access-list** command statement.

Replace *low_interface* with the lower security interface that you specified in the **static** command statement. This is the interface through which users will access the external (global) address.

The following example illustrates the three commands required to enable access to a web server with the external IP address 209.165.201.12:

```
static (inside, outside) 209.165.201.12 10.1.1.3 netmask 255.255.255.255 0 0
access-list acl_out permit tcp any host 209.165.201.12 eq www
access-group acl_out in interface outside
```

This example uses the same **static** command that was shown in the previous section.

Controlling Outbound Connections

By default, all connections initiated on a network with a higher security level are allowed out, and you configure restrictions. You can control outbound access by IP address and protocol port, or combine access control with user authentication, as described in [“Using Authentication and Authorization.”](#) If you are not enforcing restrictions on outbound network traffic, you do not need outbound access lists.

An outbound access list allows you to restrict users from starting outbound connections or allows you to restrict users from accessing specific destination address or networks. Access lists work on a first-match basis, so for outbound access lists, you need to permit first and then deny after.

For example, you could restrict some users from accessing web sites, permit others access, or restrict one or more users from accessing a specific web site. Define access restrictions with the **access-list** command, and use the **access-group** command to bind the **access-list** command statements to an interface.

When creating an outbound access list, the format for the **access-list** command is the same as shown earlier in [“Allowing Inbound Connections”](#):

```
access-list ID action protocol source_address port destination_address port
```

By default, outbound access is permitted, so you use the **deny** action to restrict access when using an outbound access list.

For example, to prevent users on the 192.168.1.0 network on the inside interface from starting connections on the outside interface and permit all others, specify the 192.168.1.0 network address as the source address and the network connected to the outside interface as the destination address. In the example that follows, the network on the outside interface is 209.165.201.0. The **access-list** and **access-group** command statements are as follows.

```
access-list acl_in deny tcp 192.168.1.0 255.255.255.224 209.165.201.0 255.255.255.224
access-list acl_in permit ip any any
access-group acl_in in interface inside
```

You can also use access lists to prevent access to a specific server. For example, if you want to restrict users on the inside interface from accessing a website at address 209.165.201.29 on the outside interface, use the following commands.

```
access-list acl_in deny tcp any host 209.165.201.29 eq www
access-group acl_in in interface inside
```

These commands let any users start connections, but not to 209.165.201.29. The **access-group** command specifies that the users are on the inside interface.



Note

If controlling outbound access in your network is an important issue, consider using the Websense filtering application, described in [“Filtering URLs with Websense.”](#)

Using Authentication and Authorization

You can use access lists to control traffic based on IP address and protocol, but to control access and use for specific users or groups, you need to use authentication and authorization. Authentication, which is the process of identifying users, is supported by the PIX Firewall for RADIUS and TACACS+ servers. Authorization identifies the specific permissions for a given user.

If you want to apply authentication and authorization when an internal (local) host initiates a connection to an external (lower security) network, enable it on the internal (higher security) interface. To set up authentication and authorization to occur when an external host initiates a connection to an internal host, enable it on the outside interface.

**Note**

If you want a host on an outside (lower security level) interface to initiate connections with a host on an internal (higher security level) interface, create **static** and **access-list** command statements for the connection.

This section includes the following topics:

- [Configuring AAA](#)
- [Configuring RADIUS Authorization](#)

Configuring AAA

To enable authentication and authorization, identify the authentication server you are using and the server encryption key on the PIX Firewall. From the configuration on the authentication server you need to determine the users that can access the network, the services that they can use, and the hosts that they can access. Once you have this information, you can configure the PIX Firewall to either enable or disable authentication or authorization.

In addition, you can configure the PIX Firewall to control user access to specific hosts or services. However, it is easier to maintain this kind of access control in a single location, at the authentication server. After you enable authentication and authorization, the PIX Firewall provides prompts inbound or outbound for users of FTP, Telnet, or HTTP (Web) access. Controlling access to a specific system or service is handled by the authentication and authorization server.

Follow these steps to enable the PIX Firewall to support TACACS+ user authentication and authorization:

- Step 1** For inbound authentication, create the **static** and **access-list** command statements required to permit outside hosts to access servers on the inside network.
- Step 2** If the external network connects to the Internet, create a global address pool of registered IP addresses. Then specify the inside hosts that can start outbound connections with the **nat** command and with the access control lists features found in the **outbound** and **apply** commands.
- Step 3** Identify the server that handles authentication or authorization using the **aaa-server** command. Create a unique server group name. For example:

```
aaa-server AuthInbound protocol tacacs+
aaa-server AuthInbound (inside) host 10.1.1.1 TheUauthKey
aaa-server AuthOutbound protocol tacacs+
aaa-server AuthOutbound (inside) host 10.1.1.2 TheUauthKey
```

The first command statement creates the AuthInbound authentication group using TACACS+ authentication. The second command statement states that the AuthInbound server is on the inside interface, that its IP address is 10.1.1.1, and the encryption key is “TheUauthKey.”

The third command statement creates the AuthOutbound authentication group using TACACS+ authentication. The fourth command statement states that the AuthOutbound server is on the inside interface, that its IP address is 10.1.1.2, and the encryption key is “TheUauthKey.”

**Note**

RADIUS authorization is provided with the **access-list** command statement as described in “[Configuring RADIUS Authorization](#).”

Step 4 Enable authentication with the **aaa authentication** command:

```
aaa authentication include ftp outbound 0 0 0 0 AuthOutbound
aaa authentication include telnet outbound 0 0 0 0 AuthOutbound
aaa authentication include http outbound 0 0 0 0 AuthOutbound
aaa authentication include ftp inbound 0 0 0 0 AuthInbound
aaa authentication include telnet inbound 0 0 0 0 AuthInbound
aaa authentication include http inbound 0 0 0 0 AuthInbound
```

The AuthInbound and AuthOutbound groups are those you specified with the **aaa-server** command.

**Note**

Be careful to apply authentication only to protocols that can be authenticated. Applying authentication using the **any** keyword will prevent protocols such as SMTP or HTTPS from passing through the PIX Firewall.

Step 5 Enable authorization with the **aaa authorization** command. PIX Firewall checks the authorization request with the AAA server, which makes the decision about what services a user can access. Use one or both of the following commands to specify outbound and inbound authorization.

```
aaa authorization include ftp outbound 0 0 0 0
aaa authorization include telnet outbound 0 0 0 0
aaa authorization include http outbound 0 0 0 0
aaa authorization include ftp inbound 0 0 0 0
aaa authorization include telnet inbound 0 0 0 0
aaa authorization include http inbound 0 0 0 0
```

You can specify port ranges for the **aaa authorization** command in the following format:

```
aaa authorization include | exclude author_service [protocol/port[-port]] inbound |
outbound | if_name local_ip local_mask foreign_ip foreign_mask
```

where:

- *author_service*—The service that PIX Firewall listens to for AAA connections. Possible values are **any**, **http**, **ftp**, or **telnet**.
- *protocol*—The protocol for which you want to authorize access. Possible values are **udp**, **tcp**, or **icmp**.
- *port*—A port value or range for which you want to authorize access.
- **inbound**, **outbound**, *if_name*—Specify whether users are authenticated and authorized on inbound or outbound connections, or for connections that arrive at a specific interface.

- *local_ip, local_mask*—Specify the IP address on the higher security level interface from which or to which access is required.
- *foreign_ip, foreign_mask*—Specify the IP address on the lower security level interface from which or to which access is required.

Configuring RADIUS Authorization

PIX Firewall allows a RADIUS server to send user group attributes to the PIX Firewall in the RADIUS authentication response message.

The administrator first defines access lists on the PIX Firewall for each user group. For example, there could be access lists for each department in an organization, sales, marketing, engineering, and so on. The administrator then lists the access list in the group profile in CiscoSecure.

After the PIX Firewall authenticates a user, it can then use the CiscoSecure **acl** attribute (attribute 11, filter-id) returned by the authentication server to identify an access list for a given user group. To maintain consistency, PIX Firewall also provides the same functionality for TACACS+.



Note

Access lists can be used with either RADIUS or TACACS but authorizing FTP, HTTP, or telnet is only possible with TACACS+

To restrict users in a department to three servers and deny everything else, the **access-list** command statements are as follows:

```
access-list eng permit ip any server1 255.255.255.255
access-list eng permit ip any server2 255.255.255.255
access-list eng permit ip any server3 255.255.255.255
access-list eng deny ip any any
```

In this example, the vendor-specific attribute string in the CiscoSecure configuration has been set to **acl=eng**. Use this field in the CiscoSecure configuration to identify the **access-list** identification name. The PIX Firewall gets the **acl=acl_ID** from CiscoSecure and extracts the ACL number from the attribute string, which it puts in a user's uauth entry. When a user tries to open a connection, PIX Firewall checks the access list in the user's uauth entry, and depending on the permit or deny status of the access list match, permits or denies the connection. When a connection is denied, PIX Firewall generates a corresponding syslog message. If there is no match, then the implicit rule is to deny.

Because the source IP of a given user can vary depending on where they are logging in from, set the source address in the **access-list** command statement to **any**, and the destination address to identify the network services to which user is permitted or denied access.

The **aaa authorization** command does not require or provide a separate RADIUS option. To enable RADIUS authorization, perform the following steps.

- Step 1** Enable RADIUS authentication with the **aaa authentication** command.
- Step 2** Create the **access-list** command statements to specify the services that hosts are authorized to use with RADIUS.
- Step 3** Configure the authentication server with the vendor-specific **acl=acl_ID** identifier to specify the **access-list** ID.

When the PIX Firewall sends a request to the authentication server, it returns the **acl=acl_ID** string, which tells PIX Firewall to use the access-list command statements to determine how RADIUS users are authorized.

Filtering Outbound Connections

ActiveX objects and Java applets are security risks for outbound connections because they can contain code to attack hosts and servers. You can disable ActiveX objects and remove Java applets with the PIX Firewall **filter** command. In addition, you can use the **filter** command to work with a Websense server to remove URLs you deem inappropriate for use at your site.

This section includes the following topics:

- [Filtering ActiveX Objects](#)
- [Filtering Java Applets](#)
- [Filtering URLs with Websense](#)

Filtering ActiveX Objects

ActiveX controls, formerly known as OLE or OCX controls, are components you can insert in a web page or other application. These controls include custom forms, calendars, or any of the extensive third-party forms for gathering or displaying information. As a technology, ActiveX creates many potential problems for the network clients including causing workstations to fail, introducing network security problems, or being used to attack servers.

The PIX Firewall ActiveX feature blocks the HTML `<object>` commands by commenting them out within the HTML web page. This functionality has been added to the **filter** command with the **activex** option.



Note

The `<object>` tag is also used for Java applets, image files, and multimedia objects, which will also be blocked by the new command.

If the `<object>` or `</object>` HTML tags split across network packets or if the code in the tags is longer than the number of bytes in the MTU, PIX Firewall cannot block the tag.

Filtering Java Applets

The **filter java** command filters out Java applets that return to the PIX Firewall from an outbound connection. The user still receives the HTML page, but the web page source for the applet is commented out so that the applet cannot execute. Use 0 for the *local_ip* or *foreign_ip* IP addresses to mean all hosts.



Note

If Java applets are known to be in `<object>` tags, use the **filter activex** command to remove them.

Examples

To specify that all outbound connections have Java applet blocking, use the following command:

```
filter java 80 0 0 0 0
```

This command specifies that the Java applet blocking applies to Web traffic on port 80 from any local host and for connections to any foreign host.

```
filter java http 192.168.3.3 255.255.255.255 0 0
```

This command prevents host 192.168.3.3 from downloading Java applets.

Filtering URLs with Websense

This section contains the following topics:

- [Filtering URLs](#)
- [Websense Filtering by Username and Group](#)
- [Websense Information](#)

Filtering URLs

The **filter url** command allows you to prevent outbound users from accessing World Wide Web URLs that you designate using the Websense filtering application.

The **allow** option to the **filter** command determines how the PIX Firewall behaves in the event that the Websense server goes offline. If you use the **allow** option with the **filter** command and the Websense server goes offline, port 80 traffic passes through the PIX Firewall without filtering. Used without the **allow** option and with the server offline, PIX Firewall stops outbound port 80 (Web) traffic until the server is back online, or if another URL server is available, passes control to the next URL server.

**Note**

With the **allow** option set, PIX Firewall now passes control to an alternate server if the Websense server goes offline.

Perform the following steps to filter URLs:

-
- Step 1** Designate a Websense server with the **url-server** command.
 - Step 2** Enable filtering with the **filter** command.
 - Step 3** If needed, improve throughput with the **url-cache** command. However, this command does not update Websense logs, which may affect Websense accounting reports. Accumulate Websense run logs before using the **url-cache** command.
 - Step 4** Use the **show url-cache stats** and the **show perfmon** commands to view run information.
-

Examples

The following example filters all outbound HTTP connections except those from the 10.0.2.54 host:

```
url-server (perimeter) host 10.0.1.1
filter url http 0 0 0 0
filter url except 10.0.2.54 255.255.255.255 0 0
```

Websense Filtering by Username and Group

The Websense Server (UFS) works with the PIX Firewall to deny users from access to web sites based on the company security policy.

Websense protocol version 4 enables group and username authentication between a host and a PIX Firewall. The PIX Firewall performs a username lookup, and then the Websense server handles URL filtering and username logging.

Websense protocol version 4 contains the following enhancements:

- URL filtering allows the PIX Firewall to check outgoing URL requests against the policy defined on the Websense server.
- Username logging tracks username, group, and domain name on the Websense server.
- Username lookup enables the PIX Firewall to use the user authentication table to map the host's IP address to the username.

Websense Information

Information on Websense is available at the following website:

<http://www.websense.com/products/about/wse/>

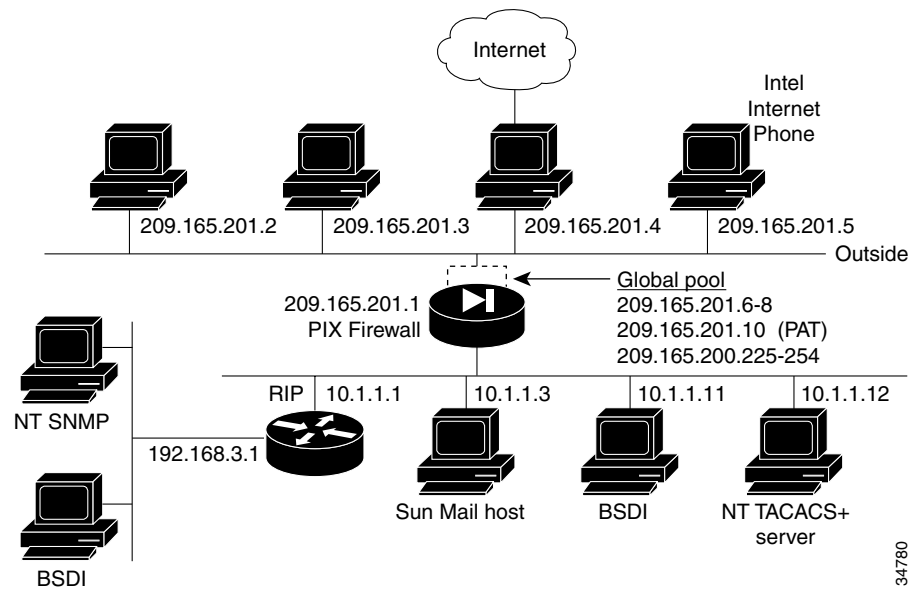
Advanced Configuration Example

This section provides a sample configuration for a PIX Firewall with two interfaces using NAT and PAT, illustrating the following types of configuration:

- [Basic Configuration](#)
- [Managing Access to Services](#)
- [Authentication and Authorization \(AAA\)](#)
- [Mail Service](#)
- [Network File System](#)

Figure 3-1 illustrates the network configuration used in this example.

Figure 3-1 Two Interfaces with NAT—Advanced



34780

Basic Configuration

The following procedure shows the basic configuration required for this example. This procedure is similar to the configuration shown in “Basic Configuration Example for Two Interfaces,” in Chapter 2, “Basic Firewall Configuration”:

Step 1 Identify the security level and names of each interface by entering the following commands:

```
nameif ethernet0 outside security0
nameif ethernet1 inside security100
```

Step 2 Identify the line speed of each interface by entering the following commands:

```
interface ethernet0 10baset
interface ethernet1 10baset
```

You may get better performance by changing the default **auto** option in the **interface** command to the specific line speed for the interface card.

Step 3 Identify the IP addresses for each interface:

```
ip address inside 10.1.1.1 255.255.255.0
ip address outside 209.165.201.1 255.255.255.224
```

Step 4 Specify the host name for the PIX Firewall:

```
hostname pixfirewall
```

This name appears in the command line prompt.

- Step 5** Let inside IP addresses be recognized on the outside network and let inside users start outbound connections:

```
nat (inside) 1 0.0.0.0 0.0.0.0
nat (inside) 2 192.168.3.0 255.255.255.0
global (outside) 1 209.165.201.6-209.165.201.8 netmask 255.255.255.224
global (outside) 1 209.165.201.10 netmask 255.255.255.224
global (outside) 2 209.165.200.225-209.165.200.254 netmask 255.255.255.224
```

- Step 6** Set the outside default route to the router attached to the Internet:

```
route outside 0 0 209.165.201.4 1
```

[Example 3-1](#) shows the listing for the basic configuration required to implement a PIX Firewall with two interfaces with NAT.

Example 3-1 Two Interfaces with NAT—Basic Configuration

```
nameif ethernet0 outside security0
nameif ethernet1 inside security100
interface ethernet0 10baset
interface ethernet1 10baset
ip address inside 10.1.1.1 255.255.255.0
ip address outside 209.165.201.1 255.255.255.224
hostname pixfirewall
nat (inside) 1 0.0.0.0 0.0.0.0
nat (inside) 2 192.168.3.0 255.255.255.0
global (outside) 1 209.165.201.6-209.165.201.8 netmask 255.255.255.224
global (outside) 1 209.165.201.10 netmask 255.255.255.224
global (outside) 2 209.165.200.225-209.165.200.254 netmask 255.255.255.224
route outside 0 0 209.165.201.4 1
```

Managing Access to Services

The following procedure shows the commands required to manage user access to H.323 and Web services. These commands are used in addition to the basic firewall configuration required, which is described in the previous section, "[Basic Configuration](#)."

- Step 1** Create outbound access lists to determine which hosts can access services:

```
access-list acl_in deny tcp host 192.168.3.3 any eq 1720
access-list acl_in deny tcp any any eq 80
access-list acl_in permit host 192.168.3.3 any eq 80
access-list acl_in permit host 10.1.1.11 any eq 80
```

The first **access-list** command statement denies host 192.168.3.3 from accessing H.323 (port 1720) services such as MS NetMeeting or InternetPhone. The next command statement denies all hosts from accessing the Web (port 80). The next command statement permits host 192.168.3.3 to use the Web. The last **access-list** command statement permits host 10.1.1.11 access to the Web (at port 80).

- Step 2** Specify that the **access-list** group regulates the activities of inside hosts starting outbound connections:

```
access-group acl_in interface inside
```

Step 3 Create static address mappings:

```
static (inside, outside) 209.165.201.16 192.168.3.16 netmask 255.255.255.240
```

The **static** command statement creates a net **static** command statement, which is a **static** command statement for a set of IP addresses, in this case for IP addresses 209.165.201.17 through 209.165.201.30.

Step 4 Enable VoIP access:

```
access-list acl_out permit tcp any host 209.165.201.16 eq h323
```

The **access-list** command statement lets users on the Internet send InternetPhone (port h323) requests to users on 192.168.3.x while addressing them as 209.165.201.x.

Step 5 Establish an externally visible IP address for Web access:

```
static (inside, outside) 209.165.201.11 10.1.1.11
access-list acl_out permit tcp any host 209.165.201.11 eq 80
```

The **static** command statement with the **access-list** command statement establishes an externally visible IP address for Web access (port 80 in the **access-list** command statement).

[Example 3-1](#) shows the command listing for configuring access to services for the network illustrated in [Figure 3-1](#).

Example 3-2 Configuring Access to Services

```
access-list acl_in deny tcp host 192.168.3.3 any eq 1720
access-list acl_in deny tcp any any eq 80
access-list acl_in permit host 192.168.3.3 any eq 80
access-list acl_in permit host 10.1.1.11 any eq 80
access-group acl_in interface inside
static (inside, outside) 209.165.201.16 192.168.3.16 netmask 255.255.255.240
access-list acl_out permit tcp any host 209.165.201.16 eq h323
static (inside, outside) 209.165.201.11 10.1.1.11
access-list acl_out permit tcp any host 209.165.201.11 eq 80
```

Authentication and Authorization (AAA)

This section describes how to implement authentication and authorization for traffic through the PIX Firewall, using a TACACS+ server. The commands used for this purpose are in addition to the basic firewall configuration required, which is described in the previous section, "[Basic Configuration](#)."

The **aaa-server** command specifies the IP address of the TACACS+ authentication server. The **aaa authentication** command statement specifies that users on network 192.168.3.0 starting FTP, HTTP, and Web connections from the inside interface be prompted for their usernames and passwords before being permitted to access the servers on other interfaces. The **aaa authorization** command statement lets the users on 192.168.3.0 access FTP, HTTP, or Telnet, and any TCP connections to anywhere as authorized by the AAA server. Even though it appears that the **aaa** commands let the PIX Firewall set security policy, the authentication server actually does the work to decide which users are authenticated and what services they can access when authentication is permitted.

Example 3-3 shows the command listing for configuring access to services for the network illustrated in Figure 3-1.

Example 3-3 Authentication and Authorization Commands

```
aaa-server TACACS+ (inside) host 10.1.1.12 1q2w3e
aaa authentication include ftp inside 192.168.3.0 255.255.255.0 0 0 TACACS+
aaa authorization include ftp inside 192.168.3.0 255.255.255.0 0 0
aaa authentication include http inside 192.168.3.0 255.255.255.0 0 0 TACACS+
aaa authorization include http inside 192.168.3.0 255.255.255.0 0 0
aaa authentication include telnet inside 192.168.3.0 255.255.255.0 0 0 TACACS+
aaa authorization include telnet inside 192.168.3.0 255.255.255.0 0 0
```

Mail Service

The following commands demonstrate how to implement a mail server for the network shown in Figure 3-1. These commands are used in addition to the basic firewall configuration required, which is described in the previous section, “Basic Configuration.”

Step 1 Provide access to the 10.1.1.3 mail server through global address 209.165.201.12:

```
static (inside, outside) 209.165.201.12 10.1.1.3
netmask 255.255.255.255 0 0
access-list acl_out permit tcp any host 209.165.201.12 eq smtp
```

The **access-list** command allows any outside host access to the static via SMTP (port 25). By default, PIX Firewall restricts all access to mail servers to RFC 821 section 4.5.1 commands of DATA, HELO, MAIL, NOOP, QUIT, RCPT, and RSET. This is implemented through the Mail Guard service, which is enabled by default (**fixup protocol smtp 25**).

Another aspect of providing access to a mail server is setting being sure that you have a DNS MX record for the static’s global address, which outside users access when sending mail to your site.

Step 2 Create access to port 113, the IDENT protocol:

```
access-list acl_out permit tcp any host 209.165.201.12 eq 113
access-group acl_out in interface outside
static (inside, outside) 209.165.201.12 10.1.1.3
netmask 255.255.255.255 0 0
access-list acl_out permit tcp any host 209.165.201.12 eq smtp
access-list acl_out permit tcp any host 209.165.201.12 eq 113
access-group acl_out in interface outside
```

If the mail server has to talk to many mail servers on the outside which connect back with the now obsolete and highly criticized IDENT protocol, use this **access-list** command statement to speed up mail transmission. The **access-group** command statement binds the **access-list** command statements to the outside interface.

Example 3-4 shows the command listing for configuring access to services for the network illustrated in Figure 3-1.

Example 3-4 Configuring Mail Server Access

```
static (inside, outside) 209.165.201.12 10.1.1.3
netmask 255.255.255.255 0 0
access-list acl_out permit tcp any host 209.165.201.12 eq smtp
access-list acl_out permit tcp any host 209.165.201.12 eq 113
access-group acl_out in interface outside
static (inside, outside) 209.165.201.12 10.1.1.3
netmask 255.255.255.255 0 0
access-list acl_out permit tcp any host 209.165.201.12 eq smtp
access-list acl_out permit tcp any host 209.165.201.12 eq 113
access-group acl_out in interface outside
```

Network File System

The following commands demonstrate how to implement Network File System (NFS) for the network shown in Figure 3-1. These commands are used in addition to the basic firewall configuration required, which is described in the previous section, “Basic Configuration”:

Step 1 Refine the accessibility of the **static** command by permitting Sun RPC over the UDP portmapper on port 111 with the **rpc** literal:

```
access-list acl_out permit udp host 209.165.201.2 host 209.165.201.11 eq rpc
```

Refer to the UNIX `/etc/rpc` file and the UNIX **rpc(3N)** command page for more information.

Once you create an **access-list** command statement for RPC, you can use the following command from outside host 209.165.201.2 to track down the activity of a PCNFSD on RPC 150001.

```
rpcinfo -u 209.165.201.11 150001
```

Another use of RPC is with the following command to see the exports of 209.165.201.11 if you want to allow mounting NFS from the outside network to the inside network:

```
showmount -e 209.165.201.11
```

Many protocols based on RPC, as well as NFS, are insecure and should be used with caution. Review your security policies carefully before permitting access to RPC.

Step 2 Permit NFS access:

```
access-list acl_out permit udp host 209.165.201.2 host 209.165.201.11 eq 2049
```

NFS access occurs at port 2049 and provides access between the outside and inside, such that host 209.165.201.2 can mount 10.1.1.11 via the global address 209.165.201.11.

Example 3-5 shows the command listing for configuring access to services for the network illustrated in Figure 3-1.

Example 3-5 Configuring NFS Access

```
access-list acl_out permit udp host 209.165.201.2 host 209.165.201.11 eq rpc
access-list acl_out permit udp host 209.165.201.2 host 209.165.201.11 eq 2049
```