Troubleshooting Connections and Resources

This chapter describes how to troubleshoot the ASA.

Testing Your Configuration

This section describes how to test connectivity for the single mode ASA or for each security context, how to ping the ASA interfaces, and how to allow hosts on one interface to ping through to hosts on another interface.

Test Basic Connectivity: Pinging Addresses

Ping is a simple command that let's you determine if a particular address is alive and responsive. The following topics explain more about the command and what types of testing you can accomplish with it.

What You Can Test Using Ping

When you ping a device, a packet is sent to the device and the device returns a reply. This process enables network devices to discover, identify, and test each other.

You can using ping to do the following tests:

- Loopback testing of two interfaces—You can initiate a ping from one interface to another on the same ASA, as an external loopback test to verify basic “up” status and operation of each interface.
- Pinging to an ASA—You can ping an interface on another ASA to verify that it is up and responding.
- Pinging through an ASA—You can ping through an intermediate ASA by pinging a device on the other side of the ASA. The packets will pass through two of the intermediate ASA’s interfaces as they go in each direction. This action performs a basic test of the interfaces, operation, and response time of the intermediate unit.
- Pinging to test questionable operation of a network device—You can ping from an ASA interface to a network device that you suspect is functioning incorrectly. If the interface is configured correctly and an echo is not received, there might be problems with the device.
- Pinging to test intermediate communications—You can ping from an ASA interface to a network device that is known to be functioning correctly. If the echo is received, the correct operation of any intermediate devices and physical connectivity is confirmed.

Choosing Between ICMP and TCP Ping

The ASA includes the traditional ping, which sends ICMP Echo Request packets and gets Echo Reply packets in return. This is the standard tool and works well if all intervening network devices allow ICMP traffic. With ICMP ping, you can ping IPv4 or IPv6 addresses, or host names.

However, some networks prohibit ICMP. If this is true of your network, you can instead use TCP ping to test network connectivity. With TCP ping, the ping sends TCP SYN packets, and considers the ping a success if it receives a SYN-ACK in response. With TCP ping, you can ping IPv4 addresses or host names, but you cannot ping IPv6 addresses.

Keep in mind that a successful ICMP or TCP ping simply means that the address you are using is alive and responding to that specific type of traffic. This means that basic connectivity is working. Other policies running on a device could prevent specific types of traffic from successfully getting through a device.

Enable ICMP

By default, you can ping from a high security interface to a low security interface. You just need to enable ICMP inspection to allow returning traffic through. If you want to ping from low to high, then you need to apply an ACL to allow traffic.

When pinging an ASA interface, any ICMP rules applied to the interface must allow Echo Request and Echo Response packets. ICMP rules are optional: if you do not configure them, all ICMP traffic to an interface is allowed.

This procedure explains all of ICMP configuration you might need to complete to enable ICMP pinging of ASA interfaces, or for pinging through an ASA.

Procedure

Step 1 Ensure ICMP rules allow Echo Request/Echo Response.

ICMP rules are optional and apply to ICMP packets sent directly to an interface. If you do not apply ICMP rules, all ICMP access is allowed. In this case, no action is required.

However, if you do implement ICMP rules, ensure that you include at least the following on each interface, replacing “inside” with the name of an interface on your device.

```
hostname(config)# icmp permit 0.0.0.0 0.0.0.0 echo inside
hostname(config)# icmp permit 0.0.0.0 0.0.0.0 echo-reply inside
```

Step 2 Ensure access rules allow ICMP.

When pinging a host through an ASA, access rules must allow ICMP traffic to leave and return. The access rule must at least allow Echo Request/Echo Reply ICMP packets. You can add these rules as global rules.

Assuming you already have access rules applied to interfaces or applied globally, simply add these rules to the relevant ACL, for example:

```
hostname(config)# access-list outside_access_in extended permit icmp any any echo
hostname(config)# access-list outside_access_in extended permit icmp any any echo-reply
```

Alternatively, just allow all ICMP:

```
hostname(config)# access-list outside_access_in extended permit icmp any any
```

If you do not have access rules, you will need to also allow the other type of traffic you want, because applying any access rules to an interface adds an implicit deny, so all other traffic will be dropped. Use the **access-group** command to apply the ACL to an interface or globally.

If you are simply adding the rule for testing purposes, you can use the **no** form of the **access-list** command to remove the rule from the ACL. If the entire ACL is simply for testing purposes, use the **no access-group** command to remove the ACL from the interface.

Step 3 Enable ICMP inspection.

ICMP inspection is needed when pinging through the ASA, as opposed to pinging an interface. Inspection allows returning traffic (that is, the Echo Reply packet) to return to the host that initiated the ping, and also ensures there is one response per packet, which prevents certain types of attack.

You can simply enable ICMP inspection in the default global inspection policy.

```
hostname(config)# policy-map global_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect icmp
```

Ping Hosts

To ping any device, you simply enter **ping** with the IP address or host name, such as **ping 10.1.1.1** or **ping www.example.com**. For TCP ping, you include the **tcp** keyword and the destination port, such as **ping tcp www.example.com 80**. That is usually the extent of any test you need to run.

Example output for a successful ping:

```
Sending 5, 100-byte ICMP Echos to out-pc, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

If the ping fails, the output indicates ? for each failed attempt, and the success rate is less than 100 percent (complete failure is 0 percent):

```
Sending 5, 100-byte ICMP Echos to 10.132.80.101, timeout is 2 seconds:
?????
Success rate is 0 percent (0/5)
```

However, you can also add parameters to control some aspects of the ping. Following are your basic options:

- ICMP ping.

  ping [*if_name*] *host* [**repeat** *count*] [**timeout** *seconds*] [**data** *pattern*] [**size** *bytes*] [**validate**]

  Where:

  - *if_name* is the name of the interface by which the host is accessible. If you do not include a name, the routing table is used to determine the interface to use.
  - *host* is the IPv4, IPv6, or host name of the host you are pinging.
  - **repeat** *count* is how many packets to send. The default is 5.
  - **timeout** *seconds* is the number of seconds for each packet to time out if no response occurs. The default is 2.
  - **data** *pattern* is the hexadecimal pattern to use in the packets sent. The default is 0xabcd.
  - **size** *bytes* is the length of the packet sent. The default is 100 bytes.
  - **validate** indicates that you want reply data validated.

- TCP ping.

  ping tcp [*if_name*] *host* [*port*] [**repeat** *count*] [**timeout** *seconds*] [**source** *host ports*]

  Where:

  - *if_name* is the interface through which the source sends the ping. If you do not include a name, the routing table is used.
  - *host* is the IPv4 address or host name of the destination you are pinging. You cannot use TCP ping with IPv6 addresses.
  - *port* is the TCP port on the host you are pinging.
  - **repeat** and **timeout** have the same meaning as above.
  - **source** *host port* indicates the source host and port for the ping. Use port 0 to get a random port.

- Interactive ping.

  ping

  By entering ping without parameters, you are prompted for interface, destination, and other parameters, including extended parameters not available as keywords. Use this method if you have need for extensive control over the ping packets.

Test ASA Connectivity Systematically

If you want to do a more systematic test of ASA connectivity, you can use the following general procedure.

Before You Begin

If you want to see the syslog messages mentioned in the procedure, enable logging (the **logging enable** command, or **Configuration > Device Management > Logging > Logging Setup** in ASDM).

Although unnecessary, you can also enable ICMP debug to see messages on the ASA console as you ping ASA interfaces from external devices (you will not see debug messages for pings that go through the ASA). We recommend that you only enable pinging and debugging messages during troubleshooting, as they can affect performance. The following example enables ICMP debugging, sets syslog messages to be sent to Telnet or SSH sessions and sends them to those sessions, and enables logging. Instead of **logging monitor debug**, you can alternately use the **logging buffer debug** command to send log messages to a buffer, and then view them later using the **show logging** command.

```
hostname(config)# debug icmp trace
hostname(config)# logging monitor debug
hostname(config)# terminal monitor
hostname(config)# logging enable
```

With this configuration, you would see something like the following for a successful ping from an external host (209.165.201.2) to the ASA outside interface (209.165.201.1):

```
hostname(config)# debug icmp trace
Inbound ICMP echo reply (len 32 id 1 seq 256) 209.165.201.1 > 209.165.201.2
Outbound ICMP echo request (len 32 id 1 seq 512) 209.165.201.2 > 209.165.201.1
Inbound ICMP echo reply (len 32 id 1 seq 512) 209.165.201.1 > 209.165.201.2
Outbound ICMP echo request (len 32 id 1 seq 768) 209.165.201.2 > 209.165.201.1
Inbound ICMP echo reply (len 32 id 1 seq 768) 209.165.201.1 > 209.165.201.2
Outbound ICMP echo request (len 32 id 1 seq 1024) 209.165.201.2 > 209.165.201.1
Inbound ICMP echo reply (len 32 id 1 seq 1024) 209.165.201.1 > 209.165.201.2
```

The output shows the ICMP packet length (32 bytes), the ICMP packet identifier (1), and the ICMP sequence number (the ICMP sequence number starts at 0, and is incremented each time that a request is sent).

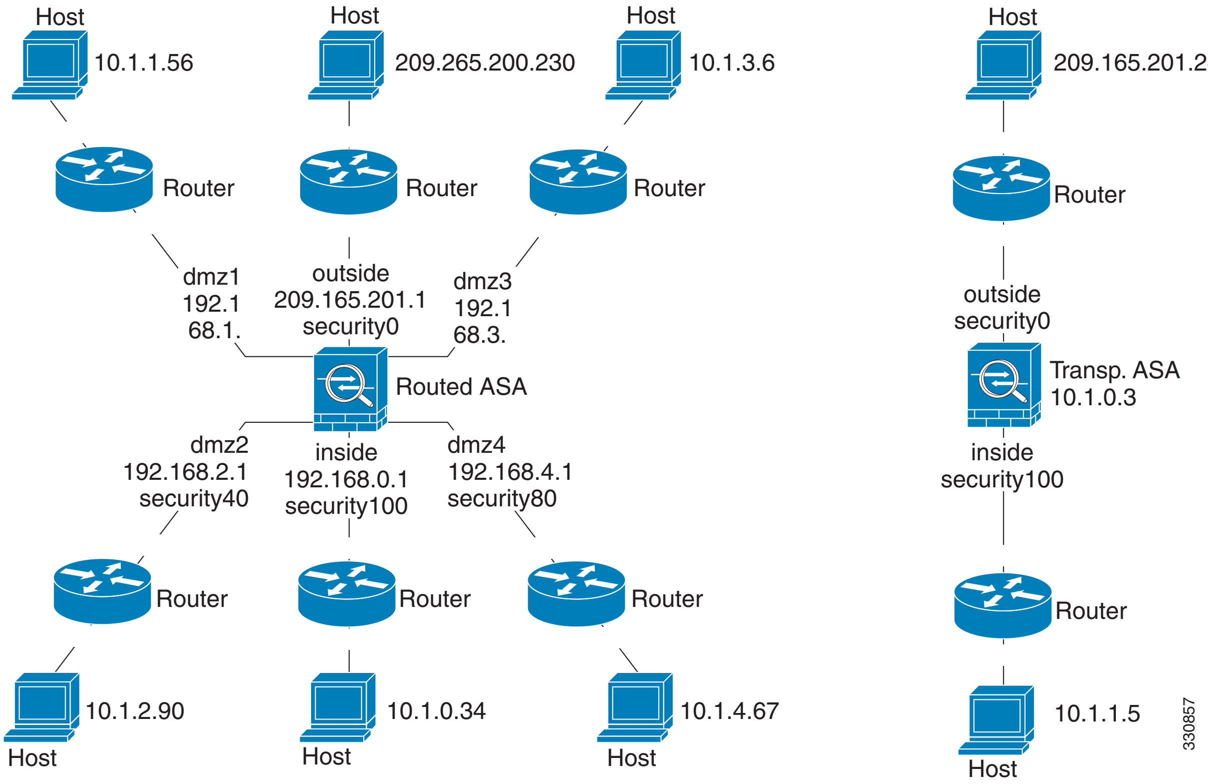
When you are finished testing, disable debugging. Leaving the configuration in place can pose performance and security risks. If you enabled logging just for testing, you can disable it also.

```
hostname(config)# no debug icmp trace
hostname(config)# no logging monitor debug
hostname(config)# terminal no monitor
hostname(config)# no logging enable
```

Procedure

Step 1 Draw a diagram of your single-mode ASA or security context that shows the interface names, security levels, and IP addresses. The diagram should also include any directly connected routers and a host on the other side of the router from which you will ping the ASA.

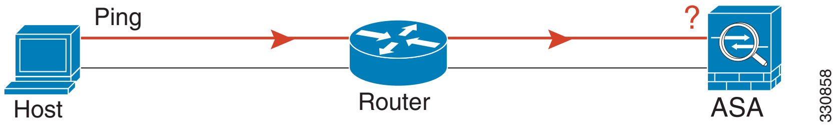
Figure 13-1 Network Diagram with Interfaces, Routers, and Hosts



Step 2 Ping each ASA interface from the directly connected routers. For transparent mode, ping the management IP address. This test ensures that the ASA interfaces are active and that the interface configuration is correct.

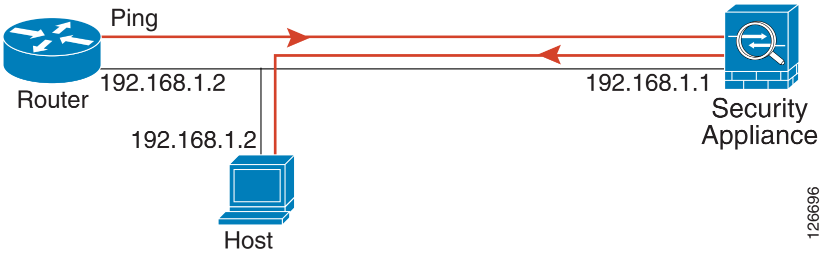
A ping might fail if the ASA interface is not active, the interface configuration is incorrect, or if a switch between the ASA and a router is down (see the following figure). In this case, no debugging messages or syslog messages appear, because the packet never reaches the ASA.

Figure 13-2 Ping Failure at the ASA Interface



If the ping reply does not return to the router, then a switch loop or redundant IP addresses might exist (see the following figure).

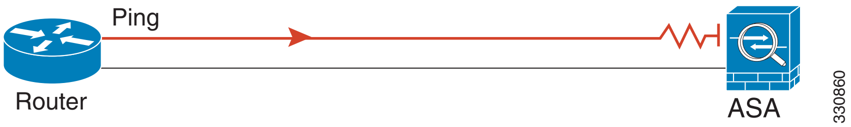
Figure 13-3 Ping Failure Because of IP Addressing Problems



Step 3 Ping each ASA interface from a remote host. For transparent mode, ping the management IP address. This test checks whether the directly connected router can route the packet between the host and the ASA, and whether the ASA can correctly route the packet back to the host.

A ping might fail if the ASA does not have a return route to the host through the intermediate router (see the following figure). In this case, the debugging messages show that the ping was successful, but syslog message 110001 appears, indicating a routing failure has occurred.

Figure 13-4 Ping Failure Because the ASA Has No Return Route



Step 4 Ping from an ASA interface to a network device that you know is functioning correctly.

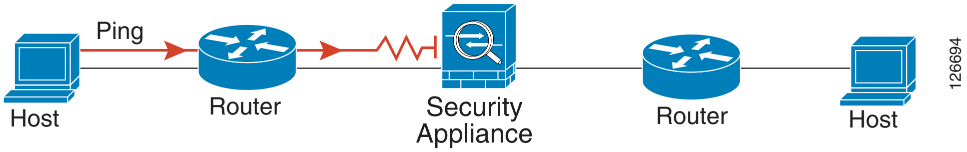
- If the ping is not received, a problem with the transmitting hardware or interface configuration may exist.
- If the ASA interface is configured correctly and it does not receive an echo reply from the “known good” device, problems with the interface hardware receiving function may exist. If a different interface with “known good” receiving capability can receive an echo after pinging the same “known good” device, the hardware receiving problem of the first interface is confirmed.

Step 5 Ping from the host or router through the source interface to another host or router on another interface. Repeat this step for as many interface pairs as you want to check. If you use NAT, this test shows that NAT is operating correctly.

If the ping succeeds, a syslog message appears to confirm the address translation for routed mode (305009 or 305011) and that an ICMP connection was established (302020). You can also enter either the **show xlate** or **show conns** command to view this information.

The ping might fail because NAT is not configured correctly. In this case, a syslog message appears, showing that the NAT failed (305005 or 305006). If the ping is from an outside host to an inside host, and you do not have a static translation, you get message 106010.

Figure 13-5 Ping Failure Because the ASA is Not Translating Addresses



Trace Routes to Hosts

If you are having problems sending traffic to an IP address, you can trace the route to the host to determine if there is a problem on the network path.

Procedure

Step 1 Make the ASA Visible on Trace Routes, page 13-8.

Step 2 Determine Packet Routes, page 13-9.

Make the ASA Visible on Trace Routes

By default, the ASA does not appear on traceroutes as a hop. To make it appear, you need to decrement the time-to-live on packets that pass through the ASA, and increase the rate limit on ICMP unreachable messages.

Procedure

Step 1 Create an L3/L4 class map to identify the traffic for which you want to customize connection settings.

```
class-map name
match parameter
```

Example:

```
hostname(config)# class-map CONNS
hostname(config-cmap)# match any
```

For information on matching statements, see Identify Traffic (Layer 3/4 Class Maps), page 1-13.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic, and identify the class map.

```
policy-map name
class name
```

Example:

```
hostname(config)# policy-map global_policy
hostname(config-pmap)# class CONNS
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name. For the class map, specify the class you created earlier in this procedure.

Step 3 Decrement time-to-live (TTL) on packets that match the class.

```
set connection decrement-ttl
```

Step 4 If you are editing an existing service policy (such as the default global policy called global_policy), you can skip this step. Otherwise, activate the policy map on one or more interfaces.

```
service-policy policymap_name {global | interface interface_name}
```

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Step 5 Increase the rate limit on ICMP Unreachable messages so that the ASA will appear on trace route output.

```
icmp unreachable rate-limit rate burst-size size
```

Example

```
hostname(config)# icmp unreachable rate-limit 50 burst-size 1
```

The rate limit can be 1-100, with 1 being the default. The burst size is meaningless, but must be 1-10.

Example

The following example decrements TTL for all traffic globally and increase the ICMP unreachable limit to 50.

```
hostname(config)# class-map global-policy
hostname(config-cmap)# match any
hostname(config-cmap)# exit
hostname(config)# policy-map global_policy
hostname(config-pmap)# class global-policy
hostname(config-pmap-c)# set connection decrement-ttl
hostname(config-pmap-c)# exit
hostname(config)# icmp unreachable rate-limit 50 burst-size 6
```

Determine Packet Routes

Use Traceroute to help you to determine the route that packets will take to their destination. A traceroute works by sending UDP packets to a destination on an invalid port. Because the port is not valid, the routers along the way to the destination respond with an ICMP Time Exceeded Message, and report that error to the ASA.

The traceroute shows the result of each probe sent. Every line of output corresponds to a TTL value in increasing order. The following table explains the output symbols.

Output Symbol	Description
*	No response was received for the probe within the timeout period.
nn msec	For each node, the round-trip time (in milliseconds) for the specified number of probes.
!N.	ICMP network unreachable.
!H	ICMP host unreachable.
!P	ICMP unreachable.
!A	ICMP administratively prohibited.
?	Unknown ICMP error.

Procedure

Step 1 Trace the route to a destination.

```
traceroute [destination_ip | hostname}
[source {source_ip | source-interface}] [numeric] [timeout timeout_value]
[probe probe_num] [ttl min_ttl max_ttl] [port port_value] [use-icmp]
```

Normally, you simply include the destination IP address or hostname, such as **traceroute www.example.com**. However, you can adjust the characteristics of the trace if desired:

- **source** {*source_ip* | *source-interface*}—Specifies the interface to use as the source of the trace. You can specify the interface by name or by IP address. In transparent mode, you must use the management address.
- **numeric**—Indicates that only the IP addresses should be shown in the trace route. Without this keyword, the trace route does DNS lookups for addresses and includes DNS names, assuming that you configure DNS.
- **timeout** *timeout_value*—How long to wait for a response before timing out. The default is 3 seconds.
- **probe** *probe_num*—How many probes to send at each TTL level. The default is 3.
- **ttl** *min_ttl max_ttl*—The minimum and maximum time-to-live values for the probes. The minimum default is one, but you can set it to a higher value to suppress the display of known hops. The maximum default is 30. The traceroute terminates when the packet reaches the destination or when the maximum value is reached.
- **port** *port_value*—The UDP port to use. The default is 33434.
- **use-icmp**—Send ICMP packets instead of UDP packets for probes.

Example

```
hostname# traceroute 209.165.200.225

Type escape sequence to abort.
Tracing the route to 209.165.200.225

1 10.83.194.1 0 msec 10 msec 0 msec
2 10.83.193.65 0 msec 0 msec 0 msec
3 10.88.193.101 0 msec 10 msec 0 msec
4 10.88.193.97 0 msec 0 msec 10 msec
5 10.88.239.9 0 msec 10 msec 0 msec
6 10.88.238.65 10 msec 10 msec 0 msec
7 172.16.7.221 70 msec 70 msec 80 msec
8 209.165.200.225 70 msec 70 msec 70 msec
```

Tracing Packets to Test Policy Configuration

You can test your policy configuration by modeling a packet based on source and destination addressing and protocol characteristics. The trace does policy lookup to test access rules, NAT, routing, and so forth, to see if the packet would be permitted or denied.

By testing packets this way, you can see the results of your policies and test whether the types of traffic you want to allow or deny are handled as desired. Besides verifying your configuration, you can use the tracer to debug unexpected behavior, such as packets being denied when they should be allowed.

Procedure

Step 1 The command is complicated, so we shall break it down into parts. Start by choosing the interface and protocol for the trace:

```
packet-tracer input ifc_name {icmp | tcp | udp | rawip} [inline-tag tag] ...
```

Where:

- **input** *ifc_name*—The name of the interface from which to start the trace.
- **icmp**, **tcp**, **udp**, **rawip**—The protocol to use. “rawip” is raw IP, that is, IP packets that are not TCP/UDP.
- **inline-tag** *tag*—(Optional.) The security group tag value embedded in the Layer 2 CMD header. Valid values range from 0 - 65533.

Step 2 Next, type in the source address and protocol criteria.

```
...{sip | user username | security-group {name name | tag tag} | fqdn fqdn-string}...
```

Where:

- *sip*—The source IPv4 or IPv6 address for the packet trace.
- **user** *username*—The user identity in the format of domain\user. The most recently mapped address for the user (if any) is used in the trace.
- **security-group** {**name** *name* | **tag** *tag*}—The source security group based on the IP-SGT lookup for Trustsec. You can specify a security group name or a tag number.
- **fqdn** *fqdn-string*—The fully qualified domain name of the source host, IPv4 only.

Step 3 Next, type in the protocol characteristics.

- ICMP—Enter the ICMP type (1-255), ICMP code (0-255), and optionally, the ICMP identifier. You must use numbers for each variable, for example, 8 for echo.

  ```
  ... type code [ident]...
  ```

- TCP/UDP—Enter the source port number.

  ```
  ... sport ...
  ```

- Raw IP—Enter the protocol number, 0-255.

  ```
  ... protocol ...
  ```

Step 4 Finally, type in the destination address criteria, destination port for TCP/UDP traces, and optional keywords, and press Enter.

```
...{dip | security-group {name name | tag tag} | fqdn fqdn-string}
dport
[detailed] [xml]
```

Where:

- *dip*—The destination IPv4 or IPv6 address for the packet trace.
- **security-group** {**name** *name* | **tag** *tag*}—The destination security group based on the IP-SGT lookup for Trustsec. You can specify a security group name or a tag number.
- **fqdn** *fqdn-string*—The fully qualified domain name of the destination host, IPv4 only.
- *dport*—The destination port for TCP/UDP traces. Do not include this value for ICMP or raw IP traces.

- **detailed**—Provides detailed trace results information in addition to the normal output.
- **xml**—Displays the trace results in XML format.

Example

The following example traces a TCP packet for the HTTP port from 10.100.10.10 to 10.100.11.11. The result indicates that the packet will be dropped by the implicit deny access rule.

```
hostname(config)# packet-tracer input outside tcp 10.100.10.10  80 10.100.11.11 80

Phase: 1
Type: ROUTE-LOOKUP
Subtype: Resolve Egress Interface
Result: ALLOW
Config:
Additional Information:
found next-hop 10.86.116.1 using egress ifc  outside

Phase: 2
Type: ACCESS-LIST
Subtype:
Result: DROP
Config:
Implicit Rule
Additional Information:

Result:
input-interface: outside
input-status: up
input-line-status: up
output-interface: NP Identity Ifc
output-status: up
output-line-status: up
Action: drop
Drop-reason: (acl-drop) Flow is denied by configured rule
```

Monitoring Performance and System Resources

You can monitor a variety of system resources to identify performance or other potential problems.

- **show perfmon**

  Shows current and average statistics for NAT xlates, connections, inspections, URL access and server requests, AAA, and TCP intercept.

- **show memory**

  Shows free and used memory.

- **show blocks**

  Shows memory block information based on block size.

- **show cpu**

  Shows CPU utilization.

- **show process**

  Shows system process information. Following are some useful variants:

- **show process cpu-usage non-zero**—Shows processes that are actually using CPU, filtering out those using 0%.
- **show process cpu-usage sorted**—Provides a breakdown of the process-related load-to-CPU that is consumed by any configured contexts.

Monitoring Connections

To view current connections with information about source, destination, protocol, and so forth, use the **show conn all detail** command.