CHAPTER 4

# Device Resiliency and Survivability

Routers and switches may be subject to attacks designed to or that indirectly affect the network availability. Possible attacks include DoS based on unauthorized and authorized protocols, Distributed DoS, flood attacks, reconnaissance, unauthorized access and more. This section presents a collection of best practices destined to preserve the resiliency and survivability of routers and switches, helping the network maintain availability even during the execution of an attack.

## CSF Methodology Assessment

The results of applying the CSF methodology are presented in table x and highlight the technologies and features identified for ensuring device resiliency and survivability, and which are integrated in Network Secure Baseline.

### Total Visibility

*Table 4-1 Device Resiliency and Survivability—Total Visibility*

| Identify | Monitor | Correlate |
|---|---|---|
| Port Security | • Logging<br>– Syslog<br>– SNMP | |

## Complete Control

*Table 2 Device Resiliency and Survivability—Complete Control*

| Harden | Isolate | Enforce |
|---|---|---|
| • Control Plane Policing<br>• Control Plane Protection<br>• Backup interfaces<br>• Element redundancy<br>– HAS<br>– RPR, RPR+<br>– SSO<br>• Standby devices<br>– Failover<br>– HSRP<br>– VRRP<br>– GLBP<br>• Topological Redundancy | | • Disable Unnecessary Services<br>• iACLs |

# Disabling Unnecessary Services

To facilitate deployment, Cisco routers and switches come out of the box with a list of services turned on that are considered appropriate for most network environments. However, since not all networks have the same requirements, some of these services may not be needed and in consequence can be disabled. Disabling these unnecessary services has two benefits. It helps preserve system resources, and eliminates the potential of security exploits on the disabled services.

This section describes how to disable some services that may not be needed. As an alternative, Cisco IOS software provides the **AutoSecure** CLI command that helps disable these unnecessary services, while enabling other security services.

**Note** Before disabling a service make sure the service is not needed.

This section describes the procedure for disabling the following services, which are typically not needed in infrastructure networks:

- Cisco Discovery Protocol (CDP)
- Directed Broadcast
- Finger
- Maintenance Operations Protocol (MOP)
- IP BOOTP Server
- IP Redirects
- IP Source Routing

- PAD
- Proxy ARP
- Ident
- TCP and UDP Small Servers

## Cisco Discovery Protocol (CDP)

Cisco Discovery Protocol (CDP) is a Cisco proprietary Layer 2 protocol designed to facilitate the administration and troubleshooting of network devices by providing information on neighboring equipment. With CDP enabled, network administrators can execute CDP commands that provide them with the platform, model, software version, and even the IP addresses of adjacent equipment.

CDP is a useful protocol, but potentially could reveal important information to an attacker. CDP is enabled by default, and can be disabled globally or for each interface. The best practice is to disable CDP globally when the service is not used, or per interface when CDP is still required. In cases where CDP is used for troubleshooting or security operations, CDP should be left enabled globally, and should be disabled only on those interfaces on which the service may represent a risk, for example, interfaces connecting to the Internet. As a general practice, CDP should not be enabled on interfaces that connect to external networks, such as the Internet.

To disable CDP globally use the **no cdp run** command from global configuration mode, as in the following example:

```
Router(config)# no cdp run
```

To disable CDP on one or more interfaces, use the **no cdp enable** command from interface configuration mode, as in the following example:

```
Router(config-if)# no cdp enable
```

**Note** Features such as ODR (on demand routing) depend on CDP, so check for dependencies prior to disabling CDP.

For more information about CDP, refer to the following URL:

http://www.cisco.com/en/US/partner/tech/tk962/technologies_tech_note09186a00801aa000.shtml

## Directed Broadcast

An IP directed broadcast packet is an IP packet whose destination address is a valid broadcast address for an IP subnet. When a directed broadcast packet reaches a router that is directly connected to its destination subnet, and if the router is configured to do so, that packet is “exploded” as a broadcast on the destination subnet. By default, earlier releases of Cisco IOS software handle directed broadcasts this way. However, because directed broadcasts have been used for attacks, such as the SMURF attack, the default behavior has been changed to drop directed broadcasts since Cisco IOS software Release 11.2.

In the case the forwarding of directed broadcast has been enabled, or in the case of Cisco IOS software releases prior to Cisco IOS software Release 11.2, it is s recommended that you disable this feature on all interfaces using the **no ip directed-broadcast** interface configuration command, as in the following example:

```
Router(config-if)# no ip directed-broadcast
```

For more information about the **ip directed-broadcast** command, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_3/ipaddr/command/reference/ip1_i1g.html#wp1081245

# Finger

Finger, as defined in RFC 742, is a protocol that can be used to obtain information about users logged into a remote host or network device. Cisco IOS software incorporates a finger service, which in Cisco IOS software releases prior to 12.1(5) and 12.1(5)T was turned on by default. Although the finger service does not reveal any extremely sensitive information, it can be used by a potential attacker to gather information. Therefore it is recommended that you disable this service.

In older releases of Cisco IOS software where the finger service was enabled by default, it can be disabled with the **no service finger** global configuration command, as in the following example:

```
Router(config)# no service finger
```

Starting in Cisco IOS software 12.1(5) and 12.1(5)T, the finger service is disabled by default. If finger has been turned on and the service is not needed, it can be disabled with the **no ip finger** global configuration command, as in the following example:

```
Router(config)# no ip finger
```

For more information on the finger service, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_3/configfun/command/reference/cfr_1g03.html#wp1033299

# Maintenance Operations Protocol (MOP)

The Maintenance Operations Protocol (MOP) was developed by Digital Equipment Corporation to be used for remote communications between hosts and servers. Cisco IOS software routers implement MOP to gather configuration information when communicating with DECNet networks. By default, MOP is enabled on all Ethernet interfaces, and disabled on all other type of interfaces. The MOP service can be disabled per interface by using the **no mop enabled** interface configuration command, as in the following example:

```
Router(config-if)# no mop enabled
```

MOP has been proven vulnerable to various attacks; therefore it should be disabled on all access and externally facing interfaces unless they provide connectivity to DECNet networks.

For more information about the **mop enabled** command, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_3/interface/command/reference/int_m1g.html#wp1139514

# IP BOOTP Server

As defined by RFC 951, the Bootstrap protocol allows a diskless workstation to configure itself at boot time by dynamically obtaining an IP address, the IP address of the BOOTP server, and a configuration file. Cisco IOS software implements a bootstrap service that allows a router to act as a BOOTP server providing dynamic configuration services to other Cisco IOS software routers. This service is turned on by default and it is used by features like AutoInstall, which simplifies or automates the configuration of Cisco devices. If not needed, this service should be disabled with the **no ip bootp server** global configuration command, as in the following example:

```
Router(config)# no ip bootp server
```

For more information about the BOOTP server service, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_3/configfun/command/reference/cfr_1g03.html#wp1031545

For more information about AutoInstall, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_2/configfun/configuration/guide/fcf002.html

# IP Redirects

By default, Cisco IOS software sends ICMP redirect messages when it is forced to resend a packet through the same interface on which it was received. By sending these redirect messages the router instructs the host the specific router to use to reach a particular destination. The ICMP redirect messages can also reveal information that can potentially be used by an attacker for discovering the network topology. Therefore, it is highly recommend that you disable this service on all access and externally facing interfaces. IP redirects can be disabled on each interface using the **no ip redirects** interface configuration command, as in the following example:

```
Router(config-if)# no ip redirects
```

For more information about the **ip redirect** command, see the following website:

http://www.cisco.com/en/US/docs/ios/12_3/ipaddr/command/reference/ip1_i2g.html#wp1081518

**Note** Previously, if the Hot Standby Router Protocol (HSRP) was configured on an interface, ICMP redirect messages were disabled by default for the interface. With Cisco IOS software Release 12.1(3)T, ICMP redirect messages are enabled by default even if HSRP is configured.

# IP Source Routing

The IP protocol supports source routing options that allow the sender of an IP packet to control the route that the datagram will take toward its ultimate destination, and generally the route that any reply will take. These options are rarely used for legitimate purposes in real networks. Some older IP implementations do not process source-routed packets properly, and it may be possible to crash machines running these implementations by sending them datagrams with source routing options. By default Cisco IOS software forwards IP packets with source routing header options. As a general best practice, IP source routing should be disabled unless strictly necessary. To have the software discard any IP packet containing a source-route option, use the **no ip source-route** global configuration command as in the following example:

```
Router(config)# no ip source-route
```

For more information about the **ip source-route** command, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_3/ipaddr/command/reference/ip1_i2g.html#wp1081830

# PAD

Cisco IOS software provides a PAD (packet assembler/disassembler) service that allows simple devices such as character-mode terminals to connect to legacy X.25 networks. With this service Cisco IOS software devices and other X.25 network equipment can establish PAD sessions. By default, the PAD service is enabled on Cisco IOS software, but it could be used to gain unauthorized or inappropriate access. Therefore, unless needed, this service should be disabled with the **no service pad** global configuration command, as in the following example:

```
Router(config)# no service pad
```

For more information about the PAD service, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_3/wan/command/reference/wan_s1g.html#wp1032441

# Proxy ARP

Proxy Address Resolution Protocol (ARP), as defined in RFC 1027, is a technique that helps machines on a subnet reach remote subnets without configuring routing or a default gateway. Proxy ARP is typically implemented on routers, and when configured, the router answers all ARP requests on the local subnet on behalf of systems some hops away.

In this model, local hosts send ARP requests for each destination for which they do not have any routing information, and the router replies with its own MAC address as the next hop. Cisco IOS software by default implements proxy ARP on all interfaces. However, unless strictly needed it should be disabled with the **no ip proxy-arp** interface configuration command, as in the following example:

```
Router(config-if)# no ip proxy-arp
```

For more information about the **ip proxy-arp** command, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_3/ipaddr/command/reference/ip1_i2g.html#wp1081466

# Ident

As defined by RFC 1413, the TCP Client Identity Protocol (Ident) is a protocol that allows a system to query the identity of a user initiating a TCP connection or a host responding to a TCP connection. When implemented, the Ident service allows a user to obtain identity information by simple connecting to a TCP port on a system, and issuing a simple text string requesting information. This clearly can yield information that could be used to attack the system. Cisco IOS software routers implement an Ident service, which is disabled by default. It is highly recommended that you do not enable this service. If the Ident service has been enabled, it can be disabled by using the **no ip identd** global configuration command, as in the following example:

```
Router(config)# no ip identd
```

For more information about the **ip identd** command, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/11_2/security/command/reference/2rauthen.html#wp2545

## TCP and UDP Small Servers

TCP and UDP small servers are daemons that typically run on Unix systems and that were designed for diagnostic purposes. Cisco IOS software also provides an implementation of UDP and TCP small servers that enables echo, chargen, daytime, and discard services. Unless strictly necessary, these services should be disabled because they can be used by a potential attacker to gather information, or to directly attack the Cisco IOS software device.

TCP and UDP small services are enabled by default on Cisco IOS software Release 11.2 and earlier. These commands are disabled by default on Cisco IOS software Software Versions 11.3 and later.

These commands may be disabled using the **no service tcp-small-servers** and **no service udp-small-servers** global configuration commands, as shown in the following example:

```
Router(config)# no service tcp-small-servers
Router(config)# no service udp-small-servers
```

For more information about TCP and UDP small servers, refer to the following URL:

http://www.cisco.com/en/US/partner/products/sw/iosswrel/ps1818/products_tech_note09186a008019d97a.shtml#tcp_udp_servers

# Infrastructure Protection Access Control Lists (iACLs)

Infrastructure protection access control lists (iACLs) is an access control technique that shields the network infrastructure from internal and external attacks. iACLs is not a feature per se, it is a technique based on extended ACLs developed initially by Internet Service Providers (ISPs) to protect their network infrastructures, but that uses concepts that can be leveraged by enterprises as well.

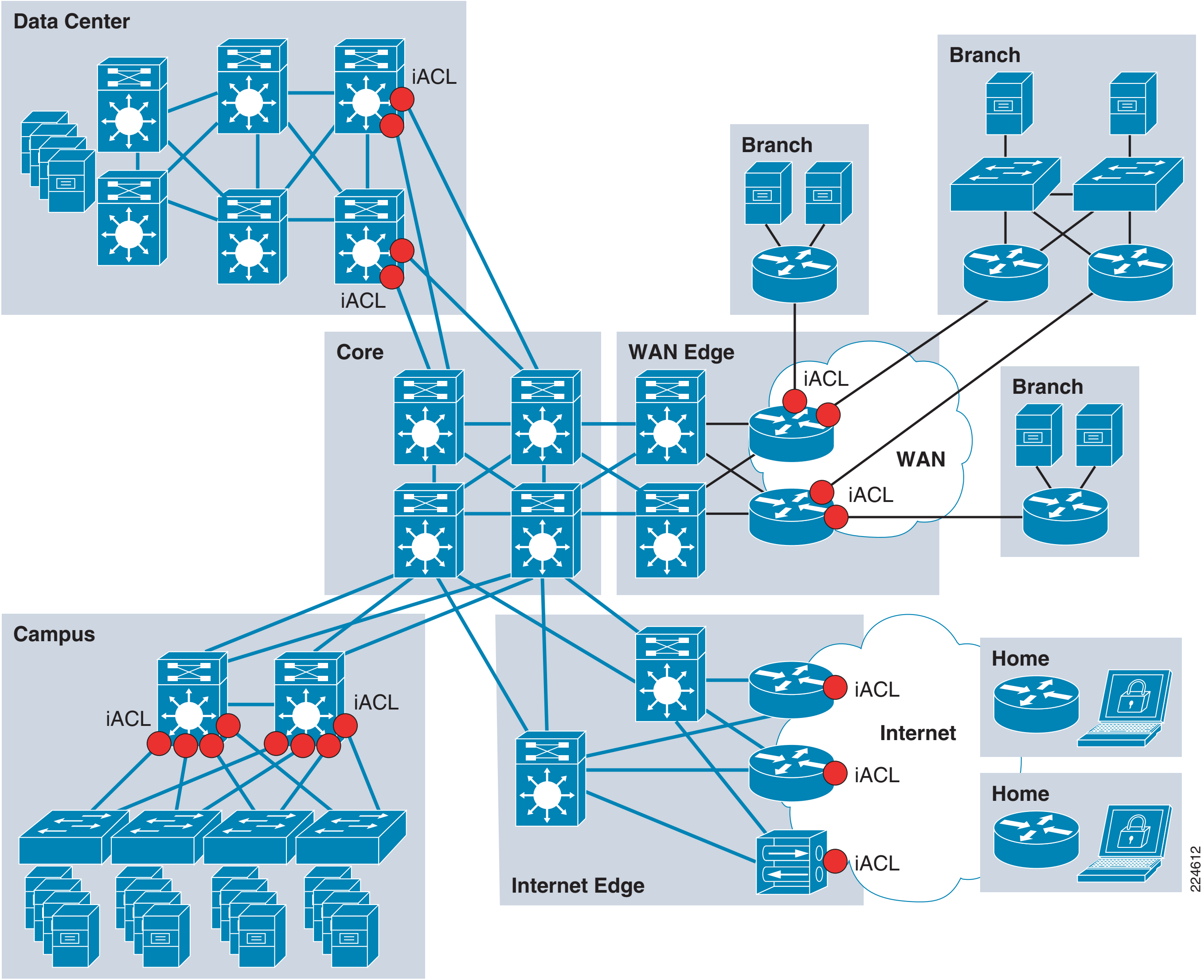
In a nutshell, iACLs are extended ACLs designed to explicitly permit authorized control and management traffic bound to the infrastructure equipment such as routers and switches, while denying any other traffic directed to the infrastructure address space. For example, an iACL deployed at an ISP peering edge is configured to explicitly permit BGP sessions from known peers, while denying any other traffic destined to the ISP's peering router as well as to the rest of the infrastructure address space.

By only allowing authorized control and management traffic, iACLs help protect routers from unauthorized access and DoS attacks based on unauthorized protocols and sources. In addition, iACLs protect routing sessions by preventing the establishment of unauthorized sessions, and by reducing the chances for session reset attacks. iACLs also help prevent the injection, modification or removal of routing information. It should be noted however, that iACLs are not effective mitigating attacks originated from trusted sources and based on trusted protocols.

iACLs are most useful when deployed at the network edges, where the infrastructure becomes accessible to internal or external users; and at administrative borders, where equipment or links under different administration meet. As mentioned in the previous example, an ISP peering edge is a good place for an iACL because it shields the ISP infrastructure from threats coming from peering links. In an enterprise, iACLs may be deployed at the many network edges.  iACLs may be deployed at the WAN edge, protecting the core infrastructure from possible threats coming from remote branch offices and partner locations. iACLs may be configured at the campus distribution, protecting the infrastructure from possible attacks originated from the LANs. Similarly, the filters deployed at the enterprise Internet edge may be designed to function as an iACL to shield the infrastructure from external threats. iACLs are also useful at administrative borders. For example, an enterprise Security Operations Center (SOC) team may decide to implement an iACL to protect its equipment from threats originated somewhere else in the enterprise, and despite the fact that iACLs may be already deployed at the WAN edge, Internet edge, campus or somewhere else. The SOC team may decide to do so simply to maintain control of the protection of its own infrastructure, rather than relying on security elements administered by other administrative teams.

Figure 4-1 shows an example deployment of iACL.

*Figure 4-1 iACL Deployment Example*



As discussed at the beginning of this document, having an adequate design of the address space facilitates deployment of security measures. This statement cannot be truer than with iACLs. As we will see in this section, the degree of summarization, the level of segmentation between the infrastructure equipment and endpoints, have a direct impact on the number of lines and complexity of the iACL. The more erratic the address space is the more complex the iACL will be and the more lines it will have.

For more information about iACLs, refer to the following URL:

http://www.cisco.com/en/US/tech/tk648/tk361/technologies_white_paper09186a00801a1a55.shtml

## iACL Structure

An iACL needs to be built in a structured manner recognizing the fact entries are processed sequentially like other ACLs. Though the specifics on how an iACL should be constructed depend on the particular deployment scenario, an iACL generally consists of four distinct modules, which are described next:

- First module—Anti-spoofing entries, entries that block packets with private addresses, special use, and other source addresses known to be invalid for the given environment,
- Second module—Entries providing explicit permission for traffic from legitimate external sources destined to infrastructure addresses. This includes control management traffic like routing protocols, remote access protocols (i.e., SSH and Telnet), network management traffic (i.e., SNMP), etc.
- Third module—**deny** statements for all other traffic from external sources destined to infrastructure addresses.
- Fourth module—**permit** statements for all other normal backbone traffic en route to non-infrastructure destinations.

The first module is designed to block any obvious illegitimate traffic, such as packets arriving with a source IP address belonging to the internal infrastructure address space, as it is an indication of spoofing. In addition, the first section of the ACL should also prevent packets with special use addresses (RFC-3330). In the case of an ISP peering iACL or enterprise internet edge iACL, the first module may also be configured to block packets arriving from the Internet with private source IP addresses (RFC-1918).

**Note** RFC 3330 defines special use addresses that might require filtering. RFC 1918 defines reserved address space that cannot be used for valid source addresses on the Internet. RFC 2827 provides ingress filtering guidelines.

The following is an example of useful entries for the first module of an iACL constructed for an ISP peering point or an enterprise Internet edge:

```
! Deny your infrastructure space as a source of external packets
access-list 101 deny ip your_infrastructure_block any
!--- Deny special-use address sources.
!--- See RFC 3330 for additional special-use addresses.
access-list 101 deny ip host 0.0.0.0 any
access-list 101 deny ip 127.0.0.0 0.255.255.255 any
access-list 110 deny ip 192.0.2.0 0.0.0.255 any
access-list 110 deny ip 224.0.0.0 31.255.255.255 any
! Deny RFC1918 space from entering AS
access-list 101 deny ip 10.0.0.0 0.255.255.255 any
access-list 101 deny ip 172.16.0.0 0.0.15.255 any
access-list 101 deny ip 192.168.0.0 0.0.255.255 any
```

The second module should allow legitimate control and management traffic (such as BGP, OSPF, SNMP, or SSH) destined to the infrastructure equipment. This requires a clear understanding of the legitimate traffic bound to the infrastructure. An iACL built without the proper understanding of the protocols and the devices involved may end up blocking critical traffic. Unless you are careful, the iACL has the potential of causing a DoS, instead of preventing it.

The following configuration fragment shows how the second section of the iACL would look for an ISP peering point or enterprise Internet edge, assuming that the only legitimate external traffic were eBGP and OSPF packets from specific peers:

```
! Permit eBGP session
access-list 101 permit tcp host bgp_peer host local_ip eq 179
```

```
access-list 101 permit tcp host bgp_peer eq 179 host local_ip
! Permit OSPF
access-list 101 permit ospf host ospf_neighbor host 224.0.0.5
! Permit DR multicast address, if needed
access-list 101 permit ospf host ospf_neighbor host 224.0.0.6
access-list 101 permit ospf host ospf_neighbor host local_ip
```

The third module of the iACL should deny any other traffic destined to the infrastructure address space, as shown in the following example:

```
! Deny all other access to infrastructure
access-list 101 deny ip any your_infrastructure_block
```

The fourth and final module of the iACL may be configured to either allow or deny all traffic, depending on the scenario. In ISP networks, which are transit networks in nature, this module should be configured to permit any other IP traffic. Likewise, in an enterprise inner iACL this module should also be configured to allow any other traffic.

```
! Permit transit traffic (ISP), enterprise inner iACL
access-list 101 permit ip any any
```

Enterprise public networks are typically the destination for traffic (not transit), and therefore the fourth module of an iACL deployed at an internet edge requires some special consideration. Depending on the existence of a firewall and the security policies in place this module may be configured to either allow or deny all other traffic. If the internet edge incorporates a firewall controlling access to the public enterprise network, then the last module of an iACL at the internet edge router may be configured to allow any other traffic (as in the example above). Per contrary,  in networks where there are no firewalls or where the Internet edge router acts as a firewall, this module may be configured to specifically permit the protocols and the IP addresses for the public services, with an implicit “deny any” denying the rest of traffic. For example:

```
! Permit only HTTP service traffic (Enterprise public services)
access-list 101 permit tcp any web_servers eq http
! access-list 101 deny ip any any (implicit)
```

This entry allows only HTTP traffic over TCP to a specific network, while an implicit *deny ip any any* denies the rest of the traffic.

## iACL Recommended Deployment Methodology

As previously mentioned, an iACL built without the proper understanding of the protocols and the devices involved may end up being ineffective and may even result in a DoS condition. For this reason, it is vital to gain an adequate level of understanding about the legitimate traffic destined to the infrastructure before deploying an iACL.

In some networks, determining the exact traffic profile needed to build the filters required might be difficult. For this reason, this document recommends a conservative methodology for deploying iACLs., using iterative ACL configurations to help identify and incrementally filter unwanted traffic as you gain a better understanding of the traffic on your network.

To deploy iACLs using this conservative methodology, complete the following steps:

**Step 1** Identify protocols used in the network using a discovery ACL.

Start by deploying a discovery or classification ACL, which permits all the commonly used protocols that access infrastructure devices.

The discovery ACL should have a source address of **any** and a destination address that encompasses the entire infrastructure IP address space. Logging can be used to develop a list of source addresses that match the protocol **permit** statements. A last line including permitting **ip any any** is required to enable traffic flow.

The objective of configuring the discovery ACL is to determine the protocols that the specific network uses. Use the **log** keyword for analysis to determine what else might be communicating with the router.

**Note** Although the **log** keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry including this keyword might result in an overwhelming number of log entries and possibly high router CPU usage. Only use the **log** keyword for short periods of time as needed to help classify traffic.

**Step 2** Review identified packets and begin filtering access to the infrastructure.

Once the packets filtered by the discovery ACL have been identified and reviewed, deploy an ACL with a permit any source to infrastructure addresses for the expected protocols.

As in Step 1, the **log** keyword can provide more information about the packets that match the **permit** entries. Using the **deny ip any** *your_ infrastructure_address_block* command at the end can help identify any unexpected packets destined to the infrastructure equipment. In case the ACL is deployed in transit networks, the last entry should be a **permit ip any any** statement to permit the flow of transit traffic. This ACL will provide basic protection and will allow network engineers to ensure that all required traffic is permitted.

**Step 3** Restrict the range of source addresses.

Once you have a clear understanding of the protocols that must be permitted, further filtering can be performed to restrict the protocols to known or authorized source addresses. For example, for an ISP you can explicitly permit external BGP neighbors or specific GRE peer addresses.

This step narrows the risk of attack without breaking any services and allows you to apply granular control to sources that access your infrastructure equipment.

**Step 4** (Optional): Limit the destination addresses within the iACL.

This final phase is meant to limit the range of destination addresses that will accept traffic for a given protocol. This helps restrict traffic more granularly.

Refer to Chapter 8, “Getting Started with Security Baseline,” for configuration examples.

# Receive Access Control Lists

Receive Access Controls Lists (rACLs) is a feature designed to protect the Route Processor (RP) on high-end routers from unnecessary traffic that could potentially affect system performance. rACLs were originally introduced for the Cisco 12000 Series Routers, but are now available on other high-end routing platforms including the Cisco 7500 Series Routers and the Cisco 10000 Series Routers.

Simply put, a rACL is an access control list that controls the traffic sent by the various line cards to the RP on distributed architectures like the Cisco 1200 Series Routers. When configured, rACLs are first created on the RP, and then pushed to the line card CPUs. When packets enter the line cards, the packets are first sent to the line card CPU. Packets requiring processing by the RP are compared against the rACL before being sent to the RP. It should be note that rACLs apply to traffic destined to the RP only, and does not affect transit traffic.

rACLs are defined as standard or extended ACLs. They typically consist of permit statements allowing the protocols and sources that are expected by the RP, and may also include deny statements explicitly blocking unwanted traffic. Like other ACLs, rACLs have an implicit **deny ip any any** at the end.

rACL help mitigate attacks directed at the RP that are intended to overwhelm its capacity. RPs always have a limited capacity to process traffic delivered from the line cards. If a high volume of data requires punting traffic to the RP, this may overwhelm the RP, resulting in a denial of service (DoS) condition. Under such circumstances the CPU of the RP struggles to keep up with the packet examination and begins dropping packets, thereby flooding the input-hold and Selective Packet Discard (SPD) queues.

Under normal circumstances, most of the traffic handled by a router is in transit over the forwarding path. Only a small portion of the traffic needs to be sent to the RP over the receive path for further analysis. Examples of traffic that is directed to the router itself, and which is handled by the RP includes the following:

- Routing protocols
- Remote access protocols, such as SSH and telnet
- Network management traffic, such as SNMP
- Other protocols, such as ICMP, or IP options, might also require processing by the RP

Because rACLs allow only authorized traffic to be sent to the RP, they help mitigate attacks directed to the RP itself. It should be noted however, that rACLs are not effective mitigating attacks originated from trusted sources and based on trusted protocols (those permitted by the rACL entries).

By protecting the RP, rACLs help ensure router and network stability during attacks. For this reason, their deployment is recommended on all routers, but particularly on those facing the Internet or other external networks. As explained next in this document, Control Plane Policing (CoPP) extends the functionality of rACLs by incorporating rate limiting. In general, CoPP is preferred over rACLs, however rACLs are simpler to deploy and may be the best fit for users looking to avoid the more complex configuration of CoPP. rACLs can also be implemented as a first step to CoPP. The guidelines for migrating from rACL to CoPP are provided in *Infrastructure Protection on Cisco IOS Software-Based Platforms* at the following URL:

http://www.cisco.com/en/US/prod/collateral/iosswrel/ps8802/ps6970/ps1838/prod_white_paper0900aecd804ac831.pdf

As discussed previously in this document, having an adequate design of the address space facilitates deployment of security measures, in particular access control mechanisms like rACLs.

For more information about rACLs, refer to the following URL:

http://www.cisco.com/en/US/tech/tk648/tk361/technologies_white_paper09186a00801a0a5e.shtml

## rACL Recommended Deployment Methodology

Because rACLs filter traffic, before deploying rACLs, it is vital to gain an adequate understanding about the legitimate traffic destined to the RP. An rACL built without the proper understanding of the protocols and the devices involved might block critical traffic, potentially creating a denial of service (DoS) condition.

In some networks, determining the exact traffic profile needed to build the filters might be difficult. For this reason, this document recommends a conservative methodology for deploying rACLs using iterative ACL configurations to help identify and eventually filter traffic.

To deploy rACLs using this methodology, complete the following steps:

**Step 1** Identify Protocols used in the network with a discovery ACL.

Start by deploying a discovery or classification rACL permitting all the commonly used protocols that access the RP. Appendix B, "Commonly Used Protocols in the Infrastructure," contains a list of commonly used protocols in the infrastructure.

This discovery rACL should have both source and destination addresses set to any. Logging can be used to develop a list of source addresses that match the protocol permit statements. In addition to the protocol permit statement, a permit any any log line at the end of the rACL can be used to identify other protocols that would be filtered by the rACL and might require access to the RP.

The objective is to determine the protocols and the specific network uses. Logging should be used for analysis to determine everything else that might be communicating with the router.

**Note** Although the log keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses the log keyword might result in an overwhelming number of log entries and possibly high router CPU utilization. Use the log keyword for short periods of time and only when needed to help classify traffic.

**Step 2** Review identified packets and begin to filter access to the RP.

Once the packets filtered by the rACL in Step 1 have been identified and reviewed, implement an rACL with a permit any any statement for each of the expected protocols.

As in Step 1, the **log** keyword can provide more information about the packets that match the permit entries. Using deny any any log at the end can help identify any unexpected packet destined to the RP. This rACL will provide basic protection and allow network engineers to ensure that all required traffic is permitted.

The objective is to test the range of protocols that need to communicate with the router without having the explicit range of IP source and destination addresses.

**Step 3** Restrict the range of source addresses.

Only allow addresses within your allocated address block as source addresses. For example, if you are using the 198.133.219.0/24 block for your network, then permit source addresses only from 198.133.219.0/24.

This step narrows the risk of attack without breaking any services. It also provides data points of devices and users from outside your address block that might be accessing your equipment. Traffic from all outside addresses will be dropped.

There may be some exceptions to the previous rule, for example an eBGP peer will require an exception because the permitted source addresses for the session will lie outside your address block.

This phase might be left on for a few days to collect data for the next phase of narrowing the rACL.

**Step 4** Narrow the rACL permit statements to only allow known authorized source addresses.

Increasingly limit the source address to only permit sources that communicate with the RP.

**Step 5** (Optional): Limit the destination addresses in the rACL.

This final phase is meant to limit the range of destination addresses that will accept traffic for a given protocol. This helps restrict traffic more granularly.

Refer to Appendix A, "Sample Configurations," for sample configurations.

.

# Control Plane Policing (CoPP)

Control Plane Policing (CoPP) is a security infrastructure feature that protects the control plane of routers and switches by enforcing QoS policies that regulate the traffic processed by the main system CPU (route or switch processor). With CoPP, these QoS policies are configured to permit, block, or rate limit the packets handled by the main CPU. This helps protects the control plane of routers and switches from a range of attacks, including reconnaissance and direct DoS.

CoPP applies to packets handled by the main CPU, referred to as control plane traffic, and which typically include the following:

- Routing protocols
- Packets destined to the local IP address of the router
- Packets from network management protocols, such as SNMP
- Interactive access protocols, such as SSH and Telnet
- Other protocols, such as ICMP, or IP options, might also require handling by the switch CPU
  - Layer 2 packets such as BPDU, CDP, DOT1X, etc.

CoPP leverages the modular QoS command-line interface (MQC) for its policy configuration. MQC allows the separation of traffic into classes, and lets the user define and apply distinct QoS policies to each class. The QoS policies can be configured to permit all packets, drop all packets, or drop only those packets exceeding a specific rate limit.

CoPP is available on a wide range of Cisco platforms, which all deliver the same basic functionality. However, CoPP has been enhanced on some platforms to leverage the benefits of the particular hardware architectures. As a result, some platforms provide advanced forms of CoPP. Non-distributed platforms implement a centralized software-based CoPP model, while some distributed platforms provide enhanced versions of CoPP: distributed and hardware-based. In addition, as a result of the hardware differences, CoPP protocol support may vary depending on the platform. This document provides a generic description of CoPP. For detailed information on how CoPP is implemented on particular platforms, refer to the list of documents provided at Appendix C, “Related Documents.”

Functionally, CoPP comes into play right after the switching or the routing decision, and before traffic is forwarded to the control plane. When CoPP is enabled, at a high level the sequence of events as follows:

**Step 1** A packet enters the router/switch configured with CoPP on the ingress port.

**Step 2** The port performs any applicable input port and QoS services.

**Step 3** The packet gets forwarded to the router/switch processor.

**Step 4** The router/switch processor makes a routing/switching decision, determining whether or not the packet is destined to the control plane.

**Step 5** Packets destined for the control plane are processed by CoPP, and are dropped or delivered to the control plane according to each traffic class policy. Packets that have other destinations are forwarded normally.

Compared with rACLs, the rate-limiting capability of CoPP makes it more effective when dealing with DoS attacks against the control plane, in particular those based on authorized protocols and sources. rACLs only permit or deny traffic, and there are scenarios in which such a binary response is not sufficient. As an example, ICMP echo requests (pings) are commonly allowed for diagnostic purposes and should be permitted when used for their intended purpose. However, a large volume of ICMP echo-requests can overwhelm the RP and may be part of a DoS attack. CoPP can effectively handle this kind of situation by enforcing rate limiting policies per traffic class. For example, using MQC, you can define a traffic class to include ICMP echo requests and then drop packets exceeding the specified rate limit for this traffic class.

In general, CoPP is recommended on all routers and switches, but particularly on those facing the Internet or other external networks. Some users may still prefer to use rACL for simplicity, or as a first step toward implementing CoPP. For guidelines on how to migrate from rACL to CoPP, refer to *Infrastructure Protection on Cisco IOS Software-Based Platforms* at the URL below.

- For more information about CoPP, see the following:

Infrastructure Protection on Cisco IOS Software-Based Platforms

http://www.cisco.com/application/pdf/en/us/guest/products/ps1838/c1244/cdccont_0900aecd804ac831.pdf

- Infrastructure Protection on Cisco Catalyst 6500 and 4500 Series Switches

http://www.cisco.com/application/pdf/en/us/guest/netsol/ns171/c649/ccmigration_09186a0080825564.pdf

# CoPP Traffic Classification

Because CoPP filters traffic, it is critical to gain an adequate level of understanding about the legitimate traffic destined to the RP or SP prior to deployment. CoPP policies built without proper understanding of the protocols, devices or required traffic rates involved may block critical traffic. This has the potential of creating a denial of service (DoS) condition. Determining the exact traffic profile needed to build the CoPP policies might be difficult in some networks. For this reason, this document describes a conservative methodology for deploying CoPP using iterative ACL configurations to help identify and to incrementally filter traffic.

Prior to developing an actual CoPP policy, required traffic must be identified and separated into different classes. Multiple classification schemes can be used, but one recommended methodology involves classifying traffic into distinct groups based on relative importance and traffic type. This section presents an example based on ten different classes, which provides great granularity and is suitable for real world environments. It is important to note that, even though you can use this example as a reference, the actual number and type of classes needed for a given network may differ and should be selected based on local requirements, security policies, and a thorough analysis of baseline traffic.

This example defines the following nine traffic classes.

## Border Gateway Protocol (BGP)

This class defines traffic that is crucial to maintaining neighbor relationships for BGP routing protocol, such as BGP keepalives and routing updates. Maintaining BGP routing protocol is crucial to maintaining connectivity within a network or to an ISP. Sites that are not running BGP would not use this class.

## Interior Gateway Protocol (IGP)

This class defines traffic that is crucial to maintaining IGP routing protocols such as Open Shortest Path First (OSPF), Enhanced Interior Gateway Routing Protocol (EIGRP), and Routing Information Protocol (RIP). Maintaining IGP routing protocols is crucial to maintaining connectivity within a network.

## Interactive Management

This class defines interactive traffic that is required for day-to-day network operations. This class would include light volume traffic used for remote network access and management. For example, telnet, Secure Shell (SSH), Network Time Protocol (NTP), Simple Network Management Protocol (SNMP) and Terminal Access Controller Access Control System (TACACS).

## File Management

This class defines high volume traffic used for software image and configuration maintenance. This class would include traffic generated for remote file transfer. For example, Trivial File Transfer Protocol (TFTP), and File Transfer Protocol (FTP).

## Reporting

This class defines traffic used for generating network performance statistics for reporting. This class would include traffic required for using Service Assurance Agent (SAA) to generate ICMP with different DSCP settings in order to report on response times within different QoS data classes.

## Monitoring

This class defines traffic used for monitoring a router. This kind of traffic should be permitted but should never be allowed to pose a risk to the router. With CoPP, this traffic can be permitted but limited to a low rate. Examples would include packets generated by ICMP echo requests (ping) and the traceroute command.

## Critical Applications

This class defines application traffic that is crucial to a specific network. The protocols that might be included in this class include generic routing encapsulation (GRE), Hot Standby Router Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), Session Initiation Protocol (SIP), Data Link Switching (DLSw), Dynamic Host Configuration Protocol (DHCP), IPSec, and Multicast traffic.

## Layer 2 Protocols

This class defines traffic used for Address Resolution Protocol (ARP). Excessive ARP packets can potentially monopolize RP resources, starving other important processes. CoPP can be used to rate limit ARP packets to prevent this. Currently, ARP is the only Layer 2 protocol that can be specifically classified using the match command.

## Undesirable

This explicitly identifies unwanted or malicious traffic that should be dropped and denied access to the RP. For example, this class could contain packets from a well-known worm. This class is particularly useful when specific traffic destined to the router should always be denied rather than be placed into a default category. Explicitly denying traffic allows you to collect rough statistics on this traffic using show commands and thereby offers some insight into the rate of denied traffic.

### Default

This class defines all remaining traffic destined to the RP that does not match any other class. MQC provides the Default class so you can specify how to treat traffic that is not explicitly associated with any other user-defined classes. It is desirable to give such traffic access to the RP but at a highly reduced rate.

With a default classification in place, statistics can be monitored to determine the rate of otherwise unidentified traffic destined to the control plane. Once this traffic is identified, further analysis can be performed to classify it. If needed, the other CoPP policy entries can be updated to account for this traffic.

## CoPP Recommended Deployment Methodology

To implement the conservative methodology recommended for deploying CoPP, complete the following steps:

**Step 1** Determine the classification scheme for your network.

Identify the known protocols that access the RP and divide them into categories using the most useful criteria for your specific network. For example, the ten categories given in the example in this section (BGP, IGP, Interactive Management, File Management, Reporting, Critical Applications, Layer 2 Protocols, Undesirable, and Default) use a combination of relative importance and traffic type. Select a scheme suited to your specific network, which may require a larger or smaller number of classes.

**Step 2** Configure classification ACLs.

Develop an ACL for each class identified in Step 1, including one for the Default class.

Configure each ACL to permit all known protocols in its class that require access to the RP. At this point, each ACL entry should have both source and destination addresses set to **any**. In addition, the ACL for the Default class should be configured with a single entry: **permit ip any any**. This will match traffic not explicitly permitted by entries in the other ACLs.

Once the ACLs have been configured, create a **class-map** for each class defined in Step 1, including one for the Default class. Then assign each ACL to its corresponding **class-map**.

**Note** In this step you should create a separate **class-map** for the default class, rather than using the **class-default** available on some platforms. Creating a separate **class-map**, and assigning a **permit ip any any** ACL, will allow you to identify traffic not yet classified as part of another class.

Each class map should then be associated with a **policy-map** that permits all traffic, regardless of classification. The policy for each class should be set as **conform-action transmit exceed-action transmit**.

**Step 3** Review the identified traffic and adjust the classification.

Ideally, the classification performed in Step 1 identified all required traffic destined to the router. However, realistically, not all required traffic will be identified prior to deployment and the **permit ip any any** entry in the class Default ACL will log a number of packet matches. Some form of analysis will be required to determine the exact nature of the unclassified packets.

Use the **show access-lists command** to see the entries in the ACLs that are in use, and to identify any additional traffic sent to the RP. To analyze the unclassified traffic you can use one of the following techniques:

- General ACL classification as described in Characterizing and Tracing Packet Floods Using Cisco Routers, available at the following URL:
- http://www.cisco.com/en/US/tech/tk59/technologies_tech_note09186a0080149ad6.shtml
- Packet analyzers
- rACLs

Once traffic has been identified, adjust the class configuration accordingly. Remove the ACL entries for those protocols that are not used. Add a **permit any any** entry for each protocol just identified.

**Step 4** Restrict a macro range of source addresses.

Refine the classification ACLs, by only allowing the full range of the allocated address block to be permitted as the source address. For example, if the network has been allocated 172.26.0.0/16, then permit source addresses from 172.26.0.0/16 where applicable.

This step provides data points for devices or users from outside the assigned address block that might be accessing the equipment. For example, an external BGP (eBGP) peer will require an additional ACL entry because the permitted source addresses for the session will lay outside the assigned address block. This phase might be left on for a few days to collect data for the next phase of narrowing the ACL entries.

**Step 5** Narrow the ACL permit statements to authorized source addresses.

Increasingly limit the source address in the classification ACLs to only permit sources that communicate with the RP. For instance, only known network management stations should be permitted to access the SNMP ports on a router.

**Step 6** Refine CoPP policies by implementing rate limiting.

Use the **show policy-map control-plane** command to collect data about the actual policies in place. Analyze the packet count and rate information and develop a rate limiting policy accordingly.

At this point, you may decide to remove the **class-map** and ACL used for the classification of default traffic. If so, you should also replace the previously defined policy for the Default class by the **class-default** policy.

Refer to Appendix A, "Sample Configurations," for sample configurations.

# Control Plane Protection (CPP)

Control Plane Protection (CPP) is a security feature that extends the policing functionality provided by the software-based Control Plane Policing (CoPP) feature.  The CoPP feature controls the rate in which control plane traffic is sent to he Route Processor in Cisco IOS software-based devices. Control Plane Protection extends this policing functionality by dividing the Control Plane into three control plane sub-interfaces and allowing the enforcement of separate rate-limiting policies. In addition, CPP incorporates port-filtering and queue-thresholding. Port-filtering is a mechanism for the early dropping of packets that are directed to closed or non-listened IOS TCP/UDP ports. Queue-thresholding is a mechanism that limits the number of packets per protocol hold in the control-plane input queue, preventing the input queue from being overwhelmed by any single protocol traffic.

CPP is a feature that extends the policing functionality of the software-based CoPP by providing an additional layer of protection to the control plane. With CPP, the first layer of protection is provided by CoPP at an aggregate level by controlling all packets destined to the control plane. Once traffic is processed by CoPP is then handled to CPP, the second layer of protection, and which divides the traffic into three categories. Each category is processed by a control plane sub-interface with independent rate-limiting policies. This dual layer of protection provides a control hierarchy that allows for finer policy definition and enforcement.

**Note** Control Plane Protection is only available on platforms that support software-based CoPP. This feature is currently not available on platforms with hardware-based or distributed CoPP.

The three control plane sub-interfaces implemented by Control Plane Protection are:

- **Control-plane host subinterface—**This interface handles all control-plane IP packets that are destined to any of the IP addresses configured on the router interfaces. Examples of traffic falling in this category include tunnel termination traffic, management traffic or routing protocols such as SSH, SNMP, BGP, OSPF, and EIGRP. All host traffic terminates on and is processed by the router.
- **Control-plane transit subinterface—**This subinterface receives all IP packets that are software switched by the route processor. This means packets that are not directly destined to the router itself but rather traffic traversing through the router and that require process switching.
- **Control-plane CEF-exception subinterface—**This control-plane subinterface receives all IP packets that are either redirected as a result of a configured input feature in the CEF packet forwarding path for process switching or directly enqueued in the control plane input queue by the interface driver (i.e. ARP, L2 Keepalives and all non-IP host traffic).

In addition, CPP enhances the protection of the control-plane host subinterface by implementing Port-filtering and Queue-thresholding. Port-filtering is a feature that can only be applied to the control-plane host subinterface and that automatically drops packets directed toward closed or non-listened UDP/TCP ports on the router. Queue-thresholding is another feature that can only be applied to the control-plane host subinterface and that limits the number of unprocessed packets per protocol, preventing the input queue from being overwhelmed by any single protocol traffic.

At a very high level the sequence of events with Control Plane Protection is as follows:

**Step 1** A packet enters the router configured with CoPP on an ingress interface.

**Step 2** The interface performs the basic input port and QoS services.

**Step 3** The packet gets forwarded to the router processor.

**Step 4** The router processor makes a routing decision, determining whether or not the packet is destined to the control plane.

**Step 5** Packets destined for the control plane are processed by Aggregate CoPP, and are dropped or forward to the Control Plane Path according to the polices for each traffic class. Packets that have other destinations are forwarded normally.

**Step 6** Packets sent to the Control Plane Path are intercepted by the Control Plane Protection traffic classifier, which classifies the packets into the corresponding control-plane subinterfaces.

**Step 7** Packets received by each control-plane subinterface are dropped or forward to the Control Plane global input queue according to the configured policies.

**Step 8** In addition, packets sent to the control-plane host subinterface can be dropped or forwarded according to the Port-filter and Queue-thresholding policies before they are sent to the global input queue.

Similar to CoPP, CPP helps protect the RP of Cisco IOS software-based routers by filtering unwanted traffic and by rate-limiting the traffic expected by the control plane. This shields the control plane from traffic that might be part of DoS or other attacks, helping maintain network stability even during attack conditions.

CPP ability to divide the control plane traffic and rate-limit each traffic type individually, gives you greater traffic control for attack mitigation. Port-filtering and Queue-thresholding also provide for a more advanced attack protection. On one hand, Port-filtering shields the RP from packets directed to closed or non-listened TCP/UDP ports, mitigating attacks attempting to spoof legitimate traffic permitted by CoPP. On the other hand, Queue-thresholding limits protocol queue usage mitigating attacks designed to overwhelm the input queue with the flooding of a single protocol.

CPP is recommended on all software-based IOS platforms, where hardware-based CoPP is not available. CPP is particularly useful on routers facing the Internet or other external networks.

## Control Plane Protection Recommended Deployment Methodology

By protecting the RP, CPP helps ensure router and ultimately network stability during an attack. For this reason, CPP should be deployed on all software-based routers as a key protection mechanism.

Because CPP filters traffic, it is critical to gain an adequate level of understanding about the legitimate traffic destined to the RP prior to deployment. CPP policies built without proper understanding of the protocols, devices or required traffic rates involved may block critical traffic. This has the potential of creating a denial of service (DoS) condition. Determining the exact traffic profile needed to build the CPP policies might be difficult in some networks. For this reason, this document describes a conservative methodology for deploying Control Plane Protection using iterative ACL configurations to help identify and to incrementally filter traffic.

Defining CPP policies requires the use of the MQC CLI following the same procedures explained for Control Plane Policing. For examples illustrating the use of MQC CLI to configure traffic classes and policies, please refer to the CoPP section of this document.

For configuration examples, refer to Appendix A, “Sample Configurations.”

# Port Security

An attacker can mount a DoS attack against infrastructure devices by using MAC flooding to cause MAC address table exhaustion, as well as other Layer 2 Content Addressable Memory (CAM) overflow attacks. This type of attack can be addressed with a Cisco feature called Port Security. Port Security helps mitigate MAC flooding and other Layer 2 CAM overflow attacks by restricting the MAC addresses that are allowed to send traffic on a particular port. Once Port Security is enabled on a port, only packets with a permitted source MAC address are allowed to pass through the port. A permitted MAC address is referred to as a secure MAC address.

Port Security builds a list of secure MAC addresses in one of two ways, configurable on a per-interface basis:

- Dynamic learning of MAC addresses
  - Defines a maximum number of MAC addresses that will be learnt and permitted on a port.
  - Useful for dynamic environments, such as at the access edge.
- Static configuration of MAC addresses
  - Defines the static MAC addresses permitted on a port.
  - Useful for static environments, such as a server farm, a lobby, or a Demilitarized Network (DMZ).

It is possible to combine these two options on a single interface by defining a maximum number of MAC addresses to be permitted, along with some static MAC addresses. In this scenario, the static MAC addresses will count towards the maximum number of MAC addresses permitted. This may be used, for instance, to ensure that only a single, statically defined host is permitted to communicate on a specific port, such as in a lobby.

A security violation occurs when either:

- The maximum number of secure MAC addresses is reached on a secure port and the source MAC address of the ingress traffic is different from any of the identified secure MAC addresses.
- Traffic with a secure MAC address that is configured or learned on one secure port appears in another secure port in the same VLAN.

In Cisco IOS, the action to be taken upon a security policy violation is configurable based on the three options shown in Table 4-3.

*Table 4-3 Actions for Security Policy Violations*

| **Port Security Violation Mode** | **Action Upon Security Violation** | **Notification Events** | **Port Re-activation Following Security Violation** |
|---|---|---|---|
| Protect | Packets with an unknown source MAC address are dropped | None | Dynamic MAC address learning resumes upon the number of learnt MAC addresses dropping below the configured maximum number. |
| Restrict[1] | Packets with an unknown source MAC address are dropped | Syslog message generated<br>Security violation counter increments<br>SNMP trap generated (if enabled) | Dynamic MAC address learning resumes upon the number of learnt MAC addresses dropping below the configured maximum number. |
| Shutdown (default)[2] | Interface is error-disabled and the port disabled | Syslog message generated<br>Security violation counter increments<br>SNMP trap generated (if enabled) | The port can only be re-activated through manual intervention[3]. |

[1] Port Security restrict mode may have a significant impact on device CPU. Note that the Port Security restrict violation mode will impact the CPU when an attack is in progress. It is thus recommended that the performance impact of this feature and its possible implications are carefully tested and considered prior to deployment

[2] Port Security shutdown mode is the default security violation action.

[3] With Port Security shutdown mode, upon occurrence of a security violation, a port can only be re-activated by either entering the global configuration **errdisable recovery cause psecure-violation** command or by shutting down and re-enabling the port with **shutdown** and **no shutdown** commands.

Typical deployment scenarios consist of:

- A dynamic environment, such as an access edge, where a port may have Port Security enabled with the maximum number of MAC addresses set to one, enabling only one MAC address to be dynamically learnt at any one time, and a protect response action.
- A static, controlled environment, such as a server farm or a lobby, where a port may have Port Security enabled with the server or lobby client MAC address statically defined and the more severe response action of shutdown.

**Note** The static configuration and administration of large numbers of MAC address can present an operational challenge that should be balanced against the security risks.

A VoIP deployment, where a port may have Port Security enabled with the maximum number of MAC addresses defined as two, since two MAC addresses are required per port, one for the workstation and one for the phone. In addition, it is generally recommended that the security violation action be set to restrict so that the port is not entirely taken down when a violation occurs. However, care should be taken due to the possible CPU impact of restrict mode.

Port Security is supported on trunk ports but requires some specific configuration rules to be followed. Trunk Port Security allows the configuration of Port Security related parameters on a per VLAN-port basis, applying policy according to a specific VLAN on a specific port.

# Port Security Configuration

In Cisco IOS, Port Security can be enabled on an interface using the command `switchport port-security`. The example below shows dynamic Port Security, restricted to two MAC addresses, being applied to an interface, with a security violation mode of restrict, such as may be deployed on a VoIP-enabled port.

```
Router(config)# interface gigabitethernet0/1
Router(config-if)# switchport port-security maximum 2
Router(config-if)# switchport port-security violation restrict
Router(config-if)# switchport port-security
```

The example below illustrates how a port can be restricted for use by only one specific host, with the defined MAC address, such as may be employed in a lobby environment.

```
Router(config)# interface gigabitethernet0/2
Router(config-if)# switchport port-security maximum 1
Router(config-if)# switchport port-security mac-address 1000.2000.3000
Router(config-if)# switchport port-security violation restrict
Router(config-if)# switchport port-security
```

An additional configuration option called ‘sticky learning’ is also available. Sticky Port Security retains learnt MAC addresses across reboots, though it is not available on all switches. When sticky learning is enabled, the interface adds all MAC addresses that are dynamically learned to the running configuration and converts these addresses to sticky secure MAC addresses.

In Cisco IOS, sticky Port Security can be enabled on an interface using the command **`switchport port-security mac-address sticky`**.

```
Router(config)# interface gigabitethernet0/1
Router(config-if)# switchport port-security mac-address sticky
```

Port Security aging is also configurable for both static and dynamic addresses, allowing the aging timers and aging types to be defined. The timer is defined in minutes and can be configured as an absolute or as an inactivity timeout.

In Cisco IOS, Port Security aging can be enabled on an interface using the command `switchport port-security aging`. The example below shows an inactivity aging time of two minutes being applied to an interface.

```
Router(config)# interface gigabitethernet0/1
Router(config-if)# switchport port-security aging time 2
Router(config-if)# switchport port-security aging type inactivity
```

## Port Security Logging

The SNMP logging of Port Security policy violations can be enabled using the following command:

```
snmp-server enable traps port-security
```

SNMP trap rate-limiting can also be enabled to reduce the load on a device during an attack using the following command:

```
snmp-server enable traps port-security trap-rate <max number of traps per second>
```

**Note** An SNMP trap will only be sent if a security policy violation mode of restrict or shutdown is enabled on an interface.

For more information on the **switchport port-security** command on the Catalyst 3560, refer to the following URL:

http://www.cisco.com/en/US/docs/switches/lan/catalyst3560/software/release/12.2_37_se/command/reference/cli3.html#wp1948361

For more information on how to configure Port Security on a Catalyst 6500 running Cisco IOS, refer to the following URL:

http://www.cisco.com/en/US/docs/switches/lan/catalyst6500/ios/12.2SXF/native/configuration/guide/port_sec.html

For more information on how to configure Port Security on Catalyst 4500 running Cisco IOS, refer to the following URL:

http://www.cisco.com/en/US/docs/switches/lan/catalyst4500/12.2/31sg/configuration/guide/port_sec.html

# Redundancy

Networks are built out of numerous hardware and software components that may fail or that may be subject to attacks. Implementing redundant designs helps eliminate single points of failure, improving the availability of the network and making it more resistant to attacks. There are different ways one can implement redundancy, from deploying simple backup interfaces up to building complete redundant topologies. Certainly, making every single component redundant is costly; therefore design redundancy where most needed and according to the unique requirements of your network.

Cisco products offer a wide range of options for redundancy:

- Backup interfaces
- Element redundancy
- Standby devices
- Topological Redundancy

## Backup Interfaces

Cisco routers and other Cisco products allow the configuration of backup interfaces. A backup interface is an interface that remains in standby mode until the primary interface fails or goes down. The backup interface can be a physical interface such as a Basic Rate Interface (BRI), or an assigned backup dialer interface to be used in a dialer pool. When in standby mode, the backup interface remains shutdown and any routes associated with it do not appear in the routing table. The backup interface is brought up when the router detects that the primary interface goes down. Backup interfaces are implemented in pairs, and are commonly used to back up ISDN BRI connections, async lines and leased lines.

One benefit of backup interfaces is that they are independent of routing protocols; hence their operation is not conditioned by routing protocol convergence, route stability and so on. However, depending on the encapsulation used, the router may not detect when an interface goes down and in consequence the backup interface may not be brought up. For example, with a Frame Relay connection, the line protocol may not go down when a particular PVC/DLCI goes down. Since the router cannot detect the failure, the backup link may not be activated.

In Cisco IOS, backup interfaces are configured with the **backup interface** command. The following example sets serial 1 as the backup line to serial 0:

```
interface serial 0
 backup interface serial 1
```

For more information on the **backup interface** command, refer to the following URL:

http://www.cisco.com/en/US/products/ps6350/products_command_reference_chapter09186a008044377b.html#wp1078001

In addition to backup interfaces, Cisco IOS offers other features useful for implementing link redundancy. Reliable Static Routing Backup Using Object Tracking is one feature that allows to reliably backup PPPoE and IPSec tunnels. This feature relies on ICMP pings to monitor the tunnel, and when the primary gateway becomes unreachable thorough the primary channel, a DDR connection is initiated from an alternative port. In this way, Reliable Static Routing Backup Using Object Tracking ensures reliable backup in the case of several catastrophic events, such as Internet circuit failure or peer device failure. In addition, this feature is compatible with both preconfigured static routes and Dynamic Host Configuration Protocol (DHCP) configurations.

For more information on *Backup Interface for Reliable Static Routing Backup Using Object Tracking*, refer to the following URL:

http://www.cisco.com/en/US/products/sw/iosswrel/ps5413/products_feature_guide09186a00801d862d.html

# Element Redundancy

Some modular platforms allow the configuration of redundant Route Processors and other critical components, helping maintain the overall system and network availability. Cisco IOS routers provide the following element redundancy features:

- **High System Availability (Cisco 7500)—**HSA allows you to install two Route Processors in a single router to improve system availability. This feature is available only on Cisco 7500 series routers. Supporting two RPs in a router provides the most basic level of increased system availability through a "cold restart" feature. A cold restart means that when one RP fails, the other RP reboots the router. Thus, the router is never in a failed state for very long, thereby increasing system availability.
- **High Availability NPE Redundancy (Cisco 7300)—**Route processing redundancy is a feature currently available on the Cisco 7304 router. With two NPE-G100s installed in a router, the feature provides the most basic level of increased system availability through a "partial bootup" feature on the standby NPE-G100. A "partial bootup" means that when the active NPE-G100 fails or a fatal error is detected on the NPE-G100, the standby NPE-G100 will complete booting and take control

of the line cards. This minimizes the time that the router is in a failed state, thereby increasing system availability. Switchover takes approximately one minute. Configuration syncing of startup configuration (only) and ROMmon environmental variables are supported.

- **Route Processor Redundancy (RPR)—** RPR is an alternative mode to HSA and allows Cisco IOS software to be booted on the standby processor prior to switchover (a "cold boot"). In RPR, the standby RP loads a Cisco IOS image at boot time and initializes itself in standby mode; however, although the startup configuration is synchronized to the standby RP, system changes are not. In the event of a fatal error on the active RP, the system switches to the standby processor, which reinitializes itself as the active processor, reads and parses the startup configuration, reloads all of the line cards, and restarts the system.
- **Route Processor Redundancy Plus—** In RPR+ mode, the standby RP is fully initialized. For RPR+ both the active RP and the standby RP must be running the same software image. The active RP dynamically synchronizes startup and the running configuration changes to the standby RP, meaning that the standby RP need not be reloaded and reinitialized (a "hot boot"). Additionally, on the Cisco 10000 and 12000 series Internet routers, the line cards are not reset in RPR+ mode. This functionality provides a much faster switchover between the processors. Information synchronized to the standby RP includes running configuration information, startup information (Cisco 7304, Cisco 7500, Cisco 10000, and Cisco 12000 series devices), and changes to the chassis state such as online insertion and removal (OIR) of hardware. Line card, protocol, and application state information is not synchronized to the standby RP.
- **Stateful Switchover (SSO)—**SSO mode provides all the functionality of RPR+ in that Cisco IOS software is fully initialized on the standby RP. In addition, SSO supports synchronization of line card, protocol, and application state information between RPs for supported features and protocols (a "hot standby"). During switchover, system control and routing protocol execution are transferred from the active to the standby RP. Switchover may be due to a manual operation (CLI-invoked) or to a software- or hardware-initiated operation (hardware or software fault induced).

For more information on how to configure High System Availability on your Cisco 7500 series routers, refer to the following URL:

http://www.cisco.com/en/US/docs/ios/12_0/configfun/configuration/guide/fc_hsa.html

For more information on how to configure High Availability NPE Redundancy on your Cisco 7300 series routers, refer to the following URL:

http://www.cisco.com/en/US/docs/routers/7300/install_and_upgrade/7304/7304_fru/7304_npe_icg/O3613h.html

For more information on how to configure Route Processor Redundancy on your Cisco 7500 series routers, refer to the following URL:

http://www.cisco.com/en/US/products/sw/iosswrel/ps1612/products_white_paper09186a00800809c8.shtml

For more information on how to configure Route Processor Redundancy Plus on your Cisco routers, refer to the following URL:

http://www.cisco.com/en/US/products/ps6350/products_configuration_guide_chapter09186a008045526b.html

For more information on how to configure Stateful Switchover on your Cisco routers, refer to the following URL:

http://www.cisco.com/en/US/products/sw/iosswrel/ps1838/products_white_paper09186a00801ce6f9.shtml

## Standby Devices

Cisco products offer a range of failover mechanisms and redundancy protocols that allow you to deploy redundant devices, increasing system and network availability. In a typical scenario, devices are deployed in pairs, but may also be set up in groups.

Failover mechanisms can be Active/Standby or Active/Active, and Stateless or Stateful:

- **Active/Standby Failover—**In this configuration two pairs of devices are deployed. One of the devices is active, and as such it handles all the network traffic. The other device remains idle in standby mode. The standby device does not process network traffic until a failure occurs on the active device. When a failover occurs, the standby device becomes the active unit. It assumes the IP and MAC addresses of the previously active unit. Because the other devices on the network do not see any changes in the IP or MAC addresses, ARP entries do not change or time out anywhere on the network.
- **Active/Active Failover—**In an Active/Active failover configuration, both devices are active and handle network traffic. Active/Active allows load balancing traffic. This type of failover typically requires the separation of traffic in different contexts, where each unit is configured as primary for some contexts and standby for others. In some platforms, Active/Active Failover can be implemented by the means of packet or session load balancing.
- **Stateless Failover—** In this mode of failover, all active connections are dropped when a failover occurs. Clients need to reestablish connections when the new active unit takes over.
- **Stateful Failover:—**In this mode, the active unit in the failover pair continually passes per-connection state information to the standby unit. After a failover occurs, the same connection information is available at the new active unit. Supported end-user applications are not required to reconnect to keep the same communication session.

For more information on how to configure Firewall Stateful Failover on your Cisco routers, refer to the following URL:

http://www.cisco.com/en/US/products/ps6441/products_feature_guide09186a00806106ea.html

For more information on how to configure IPSec Stateful Failover on your Cisco routers, refer to the following URL:

http://www.cisco.com/en/US/products/ps6550/products_white_paper09186a0080116d4c.shtml

For more information on how to configure Failover on your ASA appliances, refer to the following URL:

http://www.cisco.com/en/US/docs/security/asa/asa72/configuration/guide/failover.html

For more information on how to configure Failover on your FWSM, refer to the following URL:

http://www.cisco.com/en/US/docs/security/fwsm/fwsm23/configuration/guide/failover.html

Cisco routers also support First Hop Redundancy Protocols such as HSRP, VRRP and GLBP. These protocols are designed to allow for transparent fail-over at the first-hop IP router. With these protocols, two or more routers are set up together in a group, sharing a single IP address, the virtual IP address. The virtual IP address is configured in each end user's workstation as a default gateway address and is cached in the host's ARP cache. One router in the group is elected as the active router, and it is responsible for handling all traffic sent to the virtual IP address. In the event the active router fails, one of the standby routers in the group takes over.

First Hop Redundancy Protocols:

- **Hot Standby Router Protocol (HSRP)—**HSRP provides high network availability by providing first-hop routing redundancy for IP hosts on Ethernet, Fiber Distributed Data Interface (FDDI), Bridge-Group Virtual Interface (BVI), LAN Emulation (LANE), or Token Ring networks configured with a default gateway IP address. HSRP is used in a group of routers for selecting an active router and a standby router. In a group of router interfaces, the active router is the router of choice for routing packets; the standby router is the router that takes over when the active router fails or when preset conditions are met. When HSRP is configured on a network segment, it provides a virtual MAC address and an IP address that is shared among a group of routers running HSRP. The address of this HSRP group is referred to as the virtual IP address. One of these devices is selected by the protocol to be the active router. The active router receives and routes packets destined for the MAC address of the group. For n routers running HSRP, n + 1 IP and MAC addresses are assigned. HSRP detects when the designated active router fails, at which point a selected standby router assumes control of the MAC and IP addresses of the Hot Standby group. A new standby router is also selected at that time. HSRP uses a priority mechanism to determine which HSRP configured router is to be the default active router. Devices that are running HSRP send and receive multicast User Datagram Protocol (UDP)-based hello messages to detect router failure and to designate active and standby routers. When the active router fails to send a hello message within a configurable period of time, the standby router with the highest priority becomes the active router. The transition of packet forwarding functions between routers is completely transparent to all hosts on the network. In addition, HSRP supports MD5 algorithm authentication to protect against spoofing.
- **Virtual Router Redundancy Protocol (VRRP)—**Election protocol that dynamically assigns responsibility for one or more virtual routers to the VRRP routers on a LAN, allowing several routers on a multiaccess link to utilize the same virtual IP address. A VRRP router is configured to run the VRRP protocol in conjunction with one or more other routers attached to a LAN. In a VRRP configuration, one router is elected as the virtual router master, with the other routers acting as backups in case the virtual router master fails. The LAN clients can then be configured with the virtual router as their default gateway. The virtual router, representing a group of routers, is also known as a VRRP group. VRRP is supported on Ethernet, Fast Ethernet, BVI, and Gigabit Ethernet interfaces, and on MPLS VPNs and VLANs. You can configure VRRP in such a way that traffic to and from LAN clients can be shared by multiple routers, thereby sharing the traffic load more equitably among available routers. In addition, VRRP supports MD5 algorithm authentication to protect against spoofing.
- **Gateway Load Balancing Protocol (GLBP)—**GLBP provides automatic router backup for IP hosts configured with a single default gateway on an IEEE 802.3 LAN. Multiple first hop routers on the LAN combine to offer a single virtual first hop IP router while sharing the IP packet forwarding load. Other routers on the LAN may act as redundant GLBP routers that will become active if any of the existing forwarding routers fail. GLBP performs a similar function for the user as HSRP and VRRP. HSRP and VRRP allow multiple routers to participate in a virtual router group configured with a virtual IP address. HSRP and VRRP also allowed for load balancing by configuring multiple virtual router groups and by configuring different default gateways on the LAN clients. Unlike HSRP and VRRP, GLPB allows load balancing using a single virtual IP address, without having to configure different default gateways. To that end, the single virtual IP address is associated with multiple virtual MAC addresses. The forwarding load is shared among all routers in a GLBP group rather than being handled by a single router while the other routers stand idle. Each host is configured with the same virtual IP address, and all routers in the virtual router group participate in forwarding packets. GLBP members communicate between each other through hello messages sent every 3 seconds to the multicast address 224.0.0.102, User Datagram Protocol (UDP) port 3222 (source and destination). In addition, GLBP supports MD5 algorithm authentication to protect against spoofing.

For more information on using HSRP for Fault-Tolerant IP Routing, refer to the following URL:

http://www.cisco.com/en/US/tech/tk1330/technologies_design_guide_chapter09186a008066670b.html

For more information on using VRRP for Fault-Tolerant IP Routing, refer to the following URL:

http://www.cisco.com/en/US/products/ps6350/products_configuration_guide_chapter09186a008042fbd9.html

For more information on using GLBP for Fault-Tolerant IP Routing, refer to the following URL:

http://www.cisco.com/en/US/products/ps6350/products_configuration_guide_chapter09186a008042fb97.html

## Topological Redundancy

Building networks with redundant links and devices help increase the overall availability of the network, making it more resistant to attacks. Topological redundancy can be implemented at both the network as well as the data link level. In both cases, the redundancy strategy depends on the dynamic capabilities of the network to recover from failure. At the network level this is done by using dynamic routing protocols like EIRP and OSPF, while at the data link level this is achieved by using the spanning tree protocol.

When implementing a redundant topology, it is critical to ensure that the resulting topology is consistent with the security policies and control mechanisms in place. In particular, no redundant path must bypass any of the controls in place.

The following documents provide guidance in the design of redundant topologies:

*High Availability Campus Network Design—Routed Access Layer using EIGRP or OSPF*

http://www.cisco.com/application/pdf/en/us/guest/netsol/ns432/c649/ccmigration_09186a0080811468.pdf

*Data Center High Availability Clusters Design Guide*

http://www.cisco.com/application/pdf/en/us/guest/netsol/ns500/c649/ccmigration_09186a00807c4528.pdf

# Device Resiliency and Survivability Summary

Table 4-4 summarizes the best practices explained in this document and lists the threat they help mitigate.

*Table 4-4 Device Resiliency and Survivability Summary*

| Security Feature/Technique | Attacks Mitigated | Best Used At |
|---|---|---|
| Disabling Unnecessary Services | Unauthorized access, reconnaissance, DoS based on unauthorized protocols, amplification attacks, access control bypass | All routers and switches |
| iACL | Unauthorized access, reconnaissance, DoS based on unauthorized protocols | Internet and other network edges, administrative edges |
| rACL | Unauthorized access, reconnaissance, DoS based on unauthorized protocols | All routers, primarily at Internet edge. rACL simplicity is prefer over CoPP. |
| CoPP | Unauthorized access, reconnaissance, DoS based on unauthorized protocols, DoS based on authorized protocols, Distributed DoS | All devices that provide hardware-based CoPP |
| Control Plane Protection | Unauthorized access, reconnaissance, DoS based on unauthorized protocols, DoS based on authorized protocols, Distributed DoS | Software only IOS devices |
| Redundancy | DoS, Distributed DoS | Critical routers and switches |