GSR: Receive Access Control Lists

Document ID: 43861

Contents

Introduction

This document describes a new security feature called receive access control lists (rACLs) [1] and presents recommendations and guidelines for rACL deployments. Receive ACLs are used to increase security on Cisco 12000 routers by protecting the router's gigabit route processor (GRP) from unnecessary and potentially nefarious traffic. Receive ACLs were added as a special waiver to the maintenance throttle for Cisco IOS ® Software Release 12.0.21S2 and integrated into Cisco IOS Software Release 12.0(22)S.

GRP Protection

Data received by a gigabit switch router (GSR) can be divided into two broad categories:

- Traffic that passes through the router via the forwarding path.
- Traffic that must be sent via the receive path to the GRP for further analysis.

In normal operations, the vast majority of traffic simply flows through a GSR en route to other destinations. However, the GRP must handle certain types of data, most notably routing protocols, remote router access, and network management traffic (such as Simple Network Management Protocol [SNMP]). In addition to this traffic, other Layer 3 packets might require the processing flexibility of the GRP. These would include certain IP options and certain forms of Internet Control Message Protocol (ICMP) packets. Refer to the appendix on receive adjacencies and punted packets for additional details regarding rACLs and receive path traffic on the GSR.

A GSR has several data paths, each servicing different forms of traffic. Transit traffic is forwarded from the ingress line card (LC) to the fabric and then to the egress card for next hop delivery. In addition to the transit traffic data path, a GSR has two other paths for traffic requiring local processing: LC to LC CPU and LC to LC CPU to fabric to GRP. The following table shows the paths for several commonly used features and protocols.

| Traffic Type	Data Path
Normal (transit) Traffic	LC to fabric to LC

Routing Protocols/SSH/SNMP	LC to LC CPU to fabric to GRP
ICMP Echo (ping)	
Logging	LC to LC CPU

The route processor for the GSR has a limited capacity to process traffic delivered from the LCs destined for the GRP itself. If a high volume of data requires punting to the GRP, that traffic can overwhelm the GRP. This results in an effective denial−of−service (DoS) attack. The CPU of the GRP struggles to keep up with the packet examination and begins to drop packets, flooding the input−hold and Selective Packet Discard (SPD) queues. [2] GSRs should be protected against three scenarios, which can result from DoS attacks directed at a GRP of the router.

- Routing protocol packet loss from a normal−priority flood
- Management session (Telnet, Secure Shell [SSH], SNMP) packet loss from a normal−priority flood
- Packet loss from a spoofed high−priority flood

Potential loss of routing protocol data during a normal−priority flood is currently alleviated by static classification and the rate limiting of traffic destined to the GRP from the LCs. Unfortunately, this approach has limitations. The rate limiting for normal−priority traffic destined to the GRP is insufficient to guarantee protection to high−priority routing protocol data if an attack is delivered via several LCs. Lowering the threshold at which normal−priority data is dropped to provide such protection only exacerbates the loss of management traffic from a normal−priority flood.

As this image shows, the rACL is executed on each LC before the packet is transmitted to the GRP.

Receive Path ACL is applied on the LC's CPU.
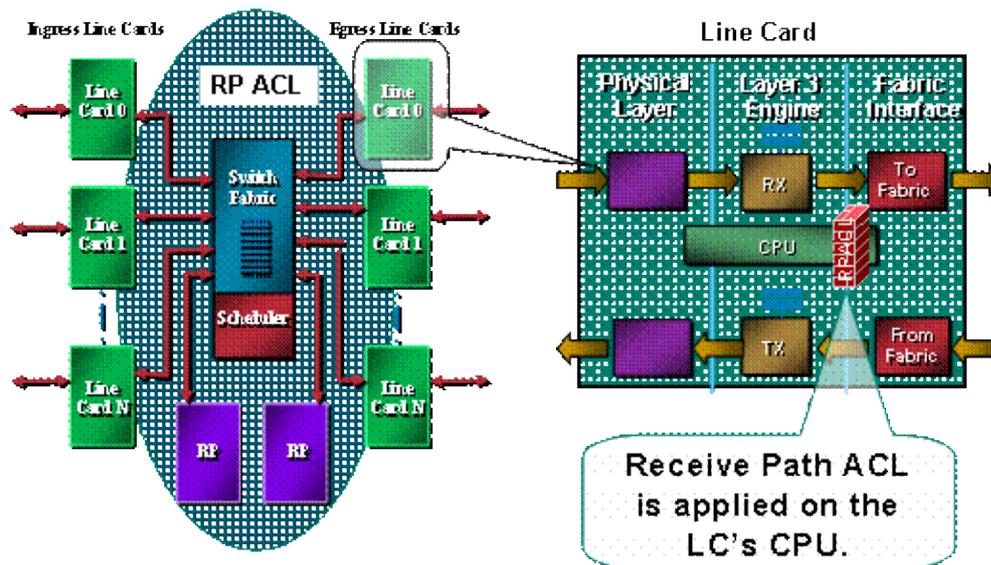
A protection mechanism for the GRP is required. rACLs affect traffic that is sent to the GRP because of receive adjacencies. Receive adjacencies are Cisco Express Forwarding adjacencies for traffic destined to the IP addresses of the router, such as the broadcast address or addresses configured on the interfaces of the router. [3] See the appendix section for more details on receive adjacencies and punted packets.

Traffic that enters an LC is first sent to the local CPU of the LC, and packets that require processing by the GRP are queued for forwarding to the route processor. The receive ACL is created on the GRP and then pushed down to the CPUs of the various LCs. Before traffic is sent from the LC CPU to the GRP, the traffic is compared to the rACL. If permitted, the traffic passes to the GRP, while all other traffic is denied. The rACL is inspected prior to the LC to GRP rate−limiting function. Since the rACL is used for all receive adjacencies, some packets that are handled by the LC CPU (such as echo requests) are subject to rACL filtering as well. This needs to be taken into account when designing rACL entries.

Receive ACLs are part one of a multipart program range of mechanisms to protect the resources in a router. Future work will include a rate–limiting component to the rACL.

Performance Impact

No memory is consumed other than that necessary to hold the single configuration entry and the defined access list itself. The rACL is copied to each LC, so a slight area of memory is taken on each LC. Overall, resources utilized are minuscule, especially when compared with the benefits of deployment.

A receive ACL does not affect the performance of forwarded traffic. The rACL only applies to receive adjacency traffic. Forwarded traffic is never subject to the rACL. Transit traffic is filtered using interface ACLs. These regular ACLs are applied to interfaces in a specified direction. Traffic is subject to ACL processing prior to rACL processing, so traffic denied by the interface ACL will not be received by the rACL. [4]

The LC performing the actual filtering (in other words, the LC receiving the traffic filtered by the rACL) will have increased CPU utilization because of the processing of the rACL. This increased CPU utilization, however, is caused by a high volume of traffic destined to the GRP; the benefit of the GRP of the rACL protection far outweighs the increased CPU utilization on an LC. The CPU utilization on an LC will vary based on LC engine type. For instance, given the same attack, an engine 3 LC will have lower CPU utilization than an engine 0 LC.

Enabling turbo ACLs (by using the **access–list compiled** command) converts ACLs into a highly efficient series of lookup table entries. When turbo ACLs are enabled, rACL depth does not affect performance. In other words, processing speed is independent of the number of entries in the ACL. If the rACL is short, turbo ACLs will not significantly increase performance but will consume memory; with short rACLs, compiled ACLs are likely not necessary.

By protecting the GRP, the rACL helps ensure router and, ultimately, network stability during an attack. As described above, the rACL is processed on the LC CPU, so the CPU utilization on each LC will increase when a large volume of data is directed at the router. On E0/E1 and some E2 bundles, CPU utilization of 100+% might lead to routing protocol and link–layer drops. These drops are localized to the card, and the GRP routing processes are protected, thus maintaining stability. E2 cards with throttling–enabled microcode [5] activate throttling mode when under heavy load and only forward precedence 6 and 7 traffic to the routing protocol. Other engine types have multi–queue architectures; for instance, E3 cards have three queues to the CPU, with routing protocol packets (precedence 6/7) in a separate, high–priority queue. High LC CPU, unless high–precedence packets cause it, will not result in routing protocol drops. Packets to the lower priority queues will be tail–dropped. Finally, E4–based cards have eight queues to the CPU, with one dedicated to routing protocol packets.

Syntax

A receive ACL is applied with the following global configuration command to distribute the rACL to each LC in the router.

```
[no] ip receive access-list <num>
```

In this syntax, *<num>* is defined as follows.

```
<1-199> IP access list (standard or extended)
<1300-2699> IP expanded access list (standard or extended)
```

Basic Template and ACL Examples

To be able to use this command, you need to define an access list that identifies traffic that should be allowed to talk to the router. The access list needs to include both routing protocols as well as management traffic (Border Gateway Protocol [BGP], Open Shortest Path First [OSPF], SNMP, SSH, Telnet). Refer to the section on deployment guidelines for more details.

The following sample ACL provides a simple outline and presents some configuration examples that can be adapted for specific uses. The ACL illustrates the required configurations for several commonly required services/protocols. For SSH, Telnet, and SNMP, a loopback address is used as the destination. For the routing protocols, the actual interface address is used. The choice of router interfaces to use in the rACL is determined by local site policies and operations. For instance, if loopbacks are used for all BGP peering sessions, then only those loopbacks need to be permitted in the **permit** statements for BGP.

```
!--- Permit BGP.

access-list 110 permit tcp host bgp_peer host loopback eq bgp

!--- Permit OSPF.

access-list 110 permit ospf host ospf_neighbor host 224.0.0.5

!--- Permit designated router multicast address, if needed.

access-list 110 permit ospf host ospf_neighbor host 224.0.0.6
access-list 110 permit ospf host ospf_neighbor host local_ip

!--- Permit Enhanced Interior Gateway Routing Protocol (EIGRP).

access-list 110 permit eigrp host eigrp_neighbor host 224.0.0.10
access-list 110 permit eigrp host eigrp_neighbor host local_ip

!--- Permit remote access by Telnet and SSH.

access-list 110 permit tcp management_addresses host loopback eq 22
access-list 110 permit tcp management_addresses host loopback eq telnet

!--- Permit SNMP.

access-list 110 permit udp host NMS_stations host loopback eq snmp

!--- Permit Network Time Protocol (NTP).

access-list 110 permit udp host ntp_server host loopback eq ntp

!--- Router-originated traceroute:
!--- Each hop returns a message that time to live (ttl)
!--- has been exceeded (type 11, code 3);
!--- the final destination returns a message that
!--- the ICMP port is unreachable (type 3, code 0).

access-list 110 permit icmp any any ttl-exceeded
access-list 110 permit icmp any any port-unreachable

!--- Permit TACACS for router authentication.

access-list 110 permit tcp host tacacs_server router_src established

!--- Permit RADIUS.

access-list 110 permit udp host radius_server router_src log
```

```
!--- Permit FTP for IOS upgrades.

access-list 110 permit tcp host image_server eq ftp host router_ip_address
access-list 110 permit tcp host image_sever eq ftp-data host router_ip_address
```

As with all Cisco ACLs, there is an implicit **deny** statement at the end of the access list, so any traffic that does not match an entry in the ACL will be denied.

Note: The **log** keyword can be used to help classify traffic destined to the GRP that is not permitted. Although the **log** keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses this keyword will increase LC CPU utilization. The performance impact associated with logging will vary with LC engine type. In general, logging should be used only when necessary on engines 0/1/2. For engines 3/4/4+, logging results in far less of an impact because of the increased CPU performance and the multi−queue architecture.

The level of granularity of this access list is determined by local security policy (for example, the level of filtering required for OSPF neighbors).

rACLS and Fragmented Packets

ACLs have a **fragments** keyword that enables specialized fragmented packet−handling behavior. In general, non−initial fragments that match the L3 statements (irrespective of the L4 information) in an ACL are affected by the **permit** or **deny** statement of the matched entry. Note that the use of the **fragments** keyword can force ACLs to either deny or permit non−initial fragments with more granularity.

In the rACL context, filtering fragments adds an additional layer of protection against a DoS attack that uses only non−initial fragments (such as FO > 0). Using a **deny** statement for non−initial fragments at the beginning of the rACL denies all non−initial fragments from accessing the router. Under rare circumstances, a valid session might require fragmentation and therefore be filtered if a **deny fragment** statement exists in the rACL.

For example, consider the partial ACL shown below.

```
access-list 110 deny tcp any any fragments
access-list 110 deny udp any any fragments
access-list 110 deny icmp any any fragments
<rest of ACL>
```

Adding these entries to the beginning of an rACL denies any non−initial fragment access to the GRP, while nonfragmented packets or initial fragments pass to the next lines of the rACL unaffected by the **deny fragment** statements. The above rACL snippet also facilitates classification of the attack since each protocol Universal Datagram Protocol (UDP), TCP, and ICMP increments separate counters in the ACL.

Refer to Access Control Lists and IP Fragments for a detailed discussion of the options.

Risk Assessment

Ensure that the rACL does not filter critical traffic such as routing protocols or interactive access to the routers. Filtering necessary traffic could result in an inability to remotely access the router, thus requiring a console connection. For this reason, lab configurations should mimic the actual deployment as closely as possible.

As always, Cisco recommends that you test this feature in the lab prior to deployment.

Appendices and Notes

Receive Adjacencies and Punted Packets

As described earlier in this document, some packets require GRP processing. The packets are punted from the data–forwarding plane to the GRP. This is a list of the common forms of Layer 3 data that require GRP access.

- Routing protocols
- Multicast control traffic (OSPF, Hot Standby Router Protocol [HSRP], Tag Distribution Protocol [TDP], Protocol Independent Multicast [PIM], and such)
- Multiprotocol Label Switching (MPLS) packets needing fragmentation
- Packets with certain IP options such as router alert
- First packet of multicast streams
- Fragmented ICMP packets that require reassembly
- All traffic destined to the router itself (except for the traffic handled on the LC)

Since rACLs apply to receive adjacencies, the rACL filters some traffic that is not punted to the GRP but is a receive adjacency. The most common example of this is an ICMP echo request (ping). ICMP echo requests directed to the router are handled by the LC CPU; since the requests are receive adjacencies, they are also filtered by the rACL. Therefore, to allow pings to the interfaces (or loopbacks) of the router, the rACL must explicitly permit the echo requests.

Receive adjacencies can be viewed using the **show ip cef** command.

```
12000-1#show ip cef
Prefix              Next Hop           Interface
0.0.0.0/0           drop               Null0 (default route handler entry)
1.1.1.1/32          attached           Null0
2.2.2.2/32          receive
64.0.0.0/30         attached           ATM4/3.300
...
```

Deployment Guidelines

Cisco recommends conservative deployment practices. To successfully deploy rACLs, the existing control and management plane access requirements must be well understood. In some networks, determining the exact traffic profile needed to build the filtering lists might be difficult. The following guidelines describe a very conservative approach for deploying rACLs using iterative rACL configurations to help identify and eventually filter traffic.

1. **Identify protocols used in the network with a classification ACL.**

   Deploy an rACL that permits all the known protocols that access the GRP. 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 that might require access to the GRP.

   The objective is to determine what protocols the specific network uses. Logging should be used 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 that uses this keyword might result in an overwhelming number of log entries and

possibly high router CPU usage. Use the **log** keyword for short periods of time and only when needed to help classify traffic.

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

   Once the packets filtered by the rACL in step 1 have been identified and reviewed, deploy an rACL with a **permit any any** statement for the allowed protocols. Just 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 packets destined to the GRP. This rACL will provide basic protection and will 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.

3. **Restrict a macro range of source addresses.**

   Only allow the full range of your allocated classless interdomain routing (CIDR) block to be permitted as the source address. For example, if you have been allocated 171.68.0.0/16 for your network, then permit source addresses from just 171.68.0.0/16.

   This step narrows the risk without breaking any services. It also provides data points of devices/people from outside your CIDR block that might be accessing your equipment. All outside address will be dropped.

   External BGP peers will require an exception, since the permitted source addresses for the session will lie outside the CIDR block.

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

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 GRP.

5. **Limit the destination addresses on the rACL. (*optional*)**

   Some Internet service providers (ISP) may choose to only allow specific protocols to use specific destination addresses on the router. This final phase is meant to limit the range of destination addresses that will accept traffic for a protocol. [6]

Deployment Example

The example below shows a receive ACL protecting a router based on the following addressing.

- The address block of the ISP is 169.223.0.0/16.
- The infrastructure block of the ISP is 169.223.252.0/22.
- The loopback for the router is 169.223.253.1/32.
- The router is a core backbone router, so only internal BGP sessions are active.

Given this information, the initial receive ACL could be something like the example below. Since we know the infrastructure address block, we will at first allow the whole block. Later, more detailed access control entries (ACEs) will be added as the specific addresses are obtained for all the devices needing access to the router.

```
!
no access-list 110
!

!--- This ACL is an explicit permit ACL.
!--- The only traffic permitted will be packets that
```

```
!--- match an explicit permit ACE.

!
! !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!--- Phase 1   Explicit Permit
!--- Permit only applications whose destination address
!--- is the loopback and whose source addresses
!--- come from an valid host.

!

!--- Note: This template must be tuned to the network s
!--- specific source address environment. Variables in
!--- the template need to be changed.

!

!--- Permit BGP.

!
access-list 110 permit tcp 169.223.252.0 0.0.3.255 host 169.223.253.1 eq bgp
!

!--- Permit OSPF.

!
access-list 110 permit ospf 169.223.252.0 0.0.3.255 host 224.0.0.5
!

!--- Permit designated router multicast address, if needed.

!
access-list 110 permit ospf 169.223.252.0 0.0.3.255 host 224.0.0.6
access-list 110 permit ospf 169.223.252.0 0.0.3.255 host 169.223.253.1
!

!--- Permit EIGRP.

!
access-list 110 permit eigrp 169.223.252.0 0.0.3.255 host 224.0.0.10
access-list 110 permit eigrp 169.223.252.0 0.0.3.255 host 169.223.253.1
!

!--- Permit remote access by Telnet and SSH.

!
access-list 110 permit tcp 169.223.252.0 0.0.3.255 host 169.223.253.1 eq 22
access-list 110 permit tcp 169.223.252.0 0.0.3.255 host 169.223.253.1 eq telnet
!

!--- Permit SNMP.

!
access-list 110 permit udp 169.223.252.0 0.0.3.255 host 169.223.253.1 eq snmp
!

!--- Permit NTP.

!
access-list 110 permit udp 169.223.252.0 0.0.3.255 host 169.223.253.1 eq ntp
!

!--- Router-originated traceroute:
!--- Each hop returns a message that ttl
!--- has been exceeded (type 11, code 3);
```

```
!--- the final destination returns a message that
!--- the ICMP port is unreachable (type 3, code 0).

!
access-list 110 permit icmp any 169.223.253.1 ttl-exceeded
access-list 110 permit icmp any 169.223.253.1 port-unreachable
!

!--- Permit TACACS for router authentication.

!
access-list 110 permit tcp 169.223.252.0 0.0.3.255 host 169.223.253.1 established
!

!--- Permit RADIUS.
!
!
access-list 110 permit udp 169.223.252.0 0.0.3.255 169.223.253.1 log
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


!--- Phase 2   Explicit Deny and Reaction
!--- Add ACEs to stop and track specific packet types
!--- that are destined for the router. This is the phase
!--- where you use ACEs with counters to track and classify attacks.

!

!--- SQL WORM Example   Watch the rate of this worm.
!--- Deny traffic destined to UDP ports 1434 and 1433.
!--- from being sent to the GRP. This is the SQL worm.

!
access-list 110 deny udp any any eq 1433
access-list 110 deny udp any any eq 1434
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!--- Phase 3   Explicit Denies for Tracking
!--- Deny all other traffic, but count it for tracking.

!
access-list 110 deny udp any any
access-list 110 deny tcp any any range 0 65535
access-list 110 deny ip any any
```

Notes

1. Refer to Understanding Selective Packet Discard (SPD) SPD and hold queue guidelines for increasing DoS resistance.
2. For more information regarding the Cisco Express Forwarding and adjacencies, refer to Cisco Express Forwarding Overview.
3. For a a detailed discussion of ACL deployment guidelines and related commands, refer to Implementing ACLs on Cisco 12000 Series Internet Routers.
4. This refers to Vanilla, Border Gateway Protocol Policy Accounting (BGPPA), Per Interface Rate Control (PIRC), and Frame Relay Traffic Policing (FRTP) bundles.
5. Phase II of the Receive Path protection will allow for the creation of a management interface, automatically limiting which IP address will listen to incoming packets.

Related Information

- **Access Lists Support Page**
- **Technical Support – Cisco Systems**

Updated: Feb 23, 2006   Document ID: 43861