

Cisco Applied Mitigation Bulletin: User Datagram Protocol Delivery Issue For IPv4/IPv6 Dual-stack Routers

<http://www.cisco.com/warp/public/707/cisco-amb-20080326-IPv4IPv6.shtml>

Revision 1.1

Last Updated 2008 April 14 1600 UTC (GMT)

For Public Release 2008 March 26 1600 UTC (GMT)

Please provide your [feedback](#) on this document.

Contents

[Cisco Response](#)
[Device Specific Mitigation and Identification](#)
[Additional Information](#)
[Revision History](#)
[Cisco Security Procedures](#)
[Related Information](#)

Cisco Response

This Applied Mitigation Bulletin is a companion document to the PSIRT Security Advisory *Cisco IOS User Datagram Protocol Delivery Issue For IPv4/IPv6 Dual-stack Routers* and provides identification and mitigation techniques that administrators can deploy on Cisco network devices.

Vulnerability Characteristics

Cisco IOS contains a vulnerability in some configurations when processing certain UDP packets over IPv6. This vulnerability can be exploited remotely without authentication and without end-user interaction. Successful exploitation of this vulnerability may prevent the interface from receiving additional traffic and lead to a denial of service (DoS) on the affected interface. Exploits that target the Resource-Reservation Protocol (RSVP) service are an exception, as they could cause a crash of the

device. Repeated attempts to exploit this vulnerability could result in a sustained DoS condition

The attack vectors for exploitation are through Internet Protocol version 6 (IPv6) packets using the following protocols and ports:

- TACACS using UDP port 49
- DNS using UDP port 53
- Resource Reservation Protocol (RSVP) using UDP port 1698
- L2TP and L2F using UDP port 1701
- IP Service Level Agreement (SLA) control protocol using UDP port 1967
- Media Gateway Control Protocol (MGCP) using UDP port 2427
- Session Initiation Protocol (SIP) using UDP port 5060

An attacker could exploit this vulnerability through spoofing attacks. This vulnerability has been assigned CVE ID CVE-2008-1153.

Information about vulnerable, unaffected, and fixed software is available in the PSIRT Security Advisory, which is available at the following link: <http://www.cisco.com/warp/public/707/cisco-sa-20080326-IPv4IPv6.shtml>.

Mitigation Technique Overview

Cisco devices provide several countermeasures for this vulnerability. Administrators are advised to consider these protection methods to be general security best practices for infrastructure devices and the traffic that transits the network.

Cisco IOS Software can provide effective means of exploit prevention using the following methods:

- IPv6 Infrastructure access control lists (iACLs)
- IPv6 Unicast Reverse Path Forwarding (Unicast RPF)

These protection mechanisms filter and drop, identify as well as verify the source IP address of, packets that are attempting to exploit the vulnerability described in this document.

The proper deployment and configuration of IPv6 Unicast RPF provides the most effective means of protection against attacks that use packets with spoofed source IPv6 addresses. IPv6 Unicast RPF should be deployed as close to all traffic sources as possible

Effective use of Cisco Embedded Event Manager (EEM) provides visibility into attacks attempting to exploit this vulnerability.

Effective means of exploit prevention can also be provided by the Cisco ASA 5500 Series Adaptive Security Appliance, the Cisco PIX 500 Series Security Appliance, and the Firewall Services Module (FWSM) for Cisco Catalyst 6500 Series switches and Cisco 7600 Series routers using the following:

- IPv6 Transit Access Control Lists (tACLs)
- Unicast RPF

These protection mechanisms filter and drop, as well as verify the source IP address of, packets that are attempting to exploit the vulnerability described in this document

Cisco IOS IPv6 NetFlow can provide visibility into these exploitation attempts using flow records.

Cisco IOS Software, Cisco ASA, Cisco PIX security appliances, and FWSM firewalls can provide visibility through syslog messages and the counter values displayed in the output from **show** commands.

Risk Management

Organizations should follow their standard risk evaluation and mitigation processes to determine the potential impact of this vulnerability. Triage refers to sorting projects and prioritizing efforts that are most likely to be successful. Cisco has provided documents that can help organizations develop a risk-based triage capability for their information security teams. [Risk Triage for Security Vulnerability Announcements](#) and [Risk Triage and Prototyping in Information Security Engagements](#) can help organizations develop repeatable security evaluation and response processes.

Device Specific Mitigation and Identification



Caution: The effectiveness of any mitigation technique is dependent on specific customer situations such as product mix, network topology, traffic behavior, and organizational mission. As with any configuration change, evaluate the impact of this configuration prior to applying the change.

Specific information about mitigation and identification is available for these devices:

- [Cisco IOS Routers and Switches](#)
- [Cisco IOS NetFlow](#)
- [Cisco ASA, PIX, and FWSM Firewalls](#)

Cisco IOS Routers and Switches

Mitigation: IPv6 Infrastructure Access Control Lists

To protect infrastructure devices and minimize the risk, impact, and effectiveness of direct infrastructure attacks, administrators are advised to deploy infrastructure access control lists (iACLs) to perform policy enforcement of traffic sent to infrastructure equipment. Administrators can construct an iACL by explicitly permitting only authorized traffic sent to infrastructure devices in accordance with existing security policies and configurations. For the maximum protection of infrastructure devices, deployed iACLs should be applied in the ingress direction on all interfaces to which an IP address has been configured.

The IPv6 iACL policy denies TACACS packets on UDP port 49, DNS packets on UDP port 53, RSVP packets on UDP port 1698, L2TP/L2F packets on UDP port 1701, IP SLA packets on UDP port 1967, MGCP packets on UDP port 2427 and SIP packets on UDP port 5060 that are sent to affected devices. In the following example, 2001:DB8:1:128::/64 is the IPv6 address space that is used by the affected devices. To mitigate against exploits of this vulnerability, all IPv6 traffic on vulnerable ports that is addressed to the affected devices must be denied. Whenever possible, infrastructure address space should be distinct from the address space used for user and services segment. Using this addressing methodology will assist with the construction and deployment of iACLs.

Additional information about iACLs is available in [Protecting Your Core: Infrastructure Protection](#)

[Access Control Lists.](#)

Note: This iACL will deny all IPv6 packets to the vulnerable UDP ports. The services will continue to operate using the IPv4 transport but will not be accessible via the IPv6 transport.

```
ipv6 access-list IPv6-Infrastructure-ACL-Policy

!-- This vulnerability does not allow trusted
!-- sources access over IPv6 on the vulnerable ports

!-- The following vulnerability-specific access control entries
!-- (ACEs) can aid in identification of attacks to global and
!-- link local addresses

deny udp any 2001:DB8:1:128::/64 eq tacacs
deny udp any 2001:DB8:1:128::/64 eq domain
deny udp any 2001:DB8:1:128::/64 eq 1698
deny udp any 2001:DB8:1:128::/64 eq 1701
deny udp any 2001:DB8:1:128::/64 eq 1967
deny udp any 2001:DB8:1:128::/64 eq 2427
deny udp any 2001:DB8:1:128::/64 eq 5060

deny udp any FE80::/10 eq tacacs
deny udp any FE80::/10 eq domain
deny udp any FE80::/10 eq 1698
deny udp any FE80::/10 eq 1701
deny udp any FE80::/10 eq 1967
deny udp any FE80::/10 eq 2427
deny udp any FE80::/10 eq 5060

!-- Permit other required traffic to the infrastructure address
!-- range and allow IPv6 Neighbor Discovery: Neighbor Solicitation and
!-- Neighbor Advertisement packets

permit icmp any any nd-ns
permit icmp any any nd-na

!-- Explicit deny for all other IP traffic to the global
!-- infrastructure address range

deny ipv6 any 2001:db8:1:128::/64

!-- Permit/deny all other Layer 3 and Layer 4 traffic
!-- in accordance with existing security policies and configurations

!-- Apply iACL to interfaces in the ingress direction

interface GigabitEthernet0/0
  ipv6 traffic-filter IPv6-Infrastructure-ACL-Policy in
```

!

Note that filtering with an interface access list will elicit the transmission of ICMP unreachable messages back to the source of the filtered traffic. Generating these messages could have the undesired effect of increasing CPU utilization on the device. In Cisco IOS Software, ICMP unreachable generation is limited to one packet every 500 milliseconds by default. ICMP unreachable message generation can be disabled using the interface configuration command **no ipv6 unreachable**. ICMP unreachable rate limiting can be changed from the default using the global configuration command **ipv6 icmp error-interval interval-in-ms [bucketsize]**.

Mitigation: Spoofing Protection Using IPv6 Unicast Reverse Path Forwarding

The vulnerability described in this document can be exploited by spoofed IPv6 packets. Administrators can deploy and configure IPv6 Unicast Reverse Path Forwarding (IPv6 Unicast RPF) as a protection mechanism against spoofing.

IPv6 Unicast RPF is configured at the interface level and can detect and drop packets that lack a verifiable source IPv6 address. Administrators should not rely on IPv6 Unicast RPF to provide 100 percent spoofing protection because spoofed packets may enter the network through a IPv6 Unicast RPF-enabled interface if an appropriate return route to the source IPv6 address exists. Administrators should take care to ensure that the appropriate IPv6 Unicast RPF mode (loose or strict) is configured during the deployment of this feature because it can drop legitimate traffic that is transiting the network. In an enterprise environment, IPv6 Unicast RPF might be enabled at the Internet edge and the internal access layer on the user-supporting Layer 3 interfaces. Loose and strict mode are indicated using the {**rx** | **any**} required choice in the **ipv6 verify unicast source reachable-via** command (where **rx** = Unicast RPF strict mode and **any** = Unicast RPF loose mode).

Configuration information for the **ipv6 verify unicast reverse-path** and **ipv6 verify unicast source reachable-via** {**rx** | **any**} commands is available in [Cisco IOS IPv6 Command Reference](#).

Identification: IPv6 Infrastructure Access Control Lists

After the administrator applies the tACL to an interface, the **show ipv6 access-lists** command will identify the number of TACACS packets on UDP port 49, DNS packets on UDP port 53, RSVP packets on UDP port 1698, L2TP/L2F packets on UDP port 1701, IP SLA packets on UDP port 1967, MGCP packets on UDP port 2427, and SIP packets on UDP port 5060 that have been filtered. Administrators should investigate filtered packets to determine whether they are attempts to exploit this vulnerability. Example output for **show ipv6 access-lists IPv6-Infrastructure-ACL-Policy** follows:

```
router#show ipv6 access-lists IPv6-Infrastructure-ACL-Policy
IPv6 access list IPv6-Infrastructure-ACL-Policy
  deny udp any 2001:DB8:1:128::/64 eq tacacs sequence 10
  deny udp any 2001:DB8:1:128::/64 eq domain (137840 matches) sequence 20
  deny udp any 2001:DB8:1:128::/64 eq 1698 sequence 30
  deny udp any 2001:DB8:1:128::/64 eq 1701 sequence 40
  deny udp any 2001:DB8:1:128::/64 eq 1967 sequence 50
  deny udp any 2001:DB8:1:128::/64 eq 2427 sequence 60
  deny udp any 2001:DB8:1:128::/64 eq 5060 (1857 matches) sequence 70
  deny udp any FE80::/10 eq tacacs sequence 80
  deny udp any FE80::/10 eq domain sequence 90
  deny udp any FE80::/10 eq 1698 sequence 100
  deny udp any FE80::/10 eq 1701 sequence 110
```

```
deny udp any FE80::/10 eq 1967 sequence 120
deny udp any FE80::/10 eq 2427 sequence 130
deny udp any FE80::/10 eq 5060 sequence 140
permit icmp any any nd-ns sequence 150
permit icmp any any nd-na sequence 160
deny ipv6 any 2001:DB8:1:128::/64 sequence 170
router#
```

In the preceding example, access list IPv6-Infrastructure-ACL-Policy has dropped the following packets received from an untrusted host or network:

- **137840 IPv6 DNS** packets on **UDP** port **53** for ACE line 20
- **1857 IPv6 SIP** packets on **UDP** port **5060** for ACE line 70

For additional information about investigating incidents using ACE counters and syslog events, reference the [Identifying Incidents Using Firewall and IOS Router Syslog Events](#) Applied Intelligence white paper.

Administrators can use Embedded Event Manager to provide instrumentation when specific conditions are met, such as ACE counter hits. The Applied Intelligence white paper [Embedded Event Manager in a Security Context](#) provides additional details about how to use this feature.

Identification: Access List Logging

The **log** or **log-input** access control list (ACL) option will cause packets that match specific ACEs to be logged. The **log-input** option enables logging of the ingress interface in addition to the packet source and destination IP addresses and ports.



Caution: Access control list logging can be very CPU intensive and must be used with extreme caution. Factors that drive the CPU impact of ACL logging are log generation, log transmission, and process switching to forward packets that match log-enabled ACEs.

For additional information about the configuration and use of ACL logging, reference the [Understanding Access Control List Logging](#) Applied Intelligence white paper.

Identification: Spoofing Protection Using IPv6 Unicast Reverse Path Forwarding

With IPv6 Unicast RPF properly deployed and configured throughout the network infrastructure, administrators can use the **show ipv6 interface**, **show cef drop**, **show cef interface type slot/port internal**, and **show ipv6 traffic** commands to identify the number of IPv6 packets that Unicast RPF for IPv6 has dropped.

Note: The **show command | begin regex**, **show command | include regex**, and **show command | section regex** command modifiers are used in the following examples to minimize the amount of output that administrators need to parse to view the desired information. Additional information about command modifiers is available in the [show command](#) sections of the Cisco IOS Configuration Fundamentals Command Reference.

Note: **show cef interface type slot/port internal** is a hidden command that must be fully entered at the command-line interface. Command completion is not available for it.

```

router#show ipv6 interface g0/0 | section RPF
  Input features: Ingress-Netflow RPF ACL
  Unicast RPF
    Process Switching:
      0 verification drops
      0 suppressed verification drops
    CEF Switching:
      25750 verification drops
      0 suppressed verification drops
router#
router#show cef drop
CEF Drop Statistics
Slot  Encap_fail  Unresolved  Unsupported  No_route  No_adj  ChkSum_Err
RP    0            0            0            0          0        0
IPv6 CEF Drop Statistics
Slot  Encap_fail  Unresolved  Unsupported  No_route  No_adj
RP    16164     0            0            25750     0
router#
router#show cef interface GigabitEthernet 0/0 internal | begin IPv6 unicast RPF
  IPv6 unicast RPF: acl=None, drop=25750, sdrop=0
  IPv6: enabled 1 unreachable TRUE redirect TRUE mtu 1500 flags 0x0
    Switching mode is CEF

    Input features: Ingress-Netflow RPF ACL
    Output features: Post-Ingress-Netflow Egress-Netflow
    Inbound access list: Infrastructure-ACL-Policy
router#
router#show ipv6 traffic | include RPF
  25750 unicast RPF drop, 0 suppressed RPF drop
router#

```

In the preceding **show cef drop** and **show ipv6 traffic** examples, IPv6 Unicast RPF has dropped **25750** IPv6 packets received globally on all interfaces with Unicast RPF configured because of the inability to verify the source address of the IPv6 packets within the Cisco Express Forwarding Information Base.

Identification: Embedded Event Manager

A Tcl-based EEM policy can be used on a vulnerable IOS device to detect the interface input blockage ("wedge") condition caused by this vulnerability by directly monitoring the hold input queue counter in all interfaces in the vulnerable device. Once EEM has detected potential exploitation of this vulnerability, the EEM policy can trigger a response by sending a syslog message or a Simple Network Management Protocol (SNMP) trap. The example EEM policy provided in this document is based on a Tcl script that monitors the interface status at defined intervals and only produces a syslog message when the hold input queue of any interface in the device reaches its maximum configured value.

The identification is based on output of the **show interfaces** command. A regular expression (Tcl **regexp** command) is utilized to parse the output of this command. The regular expression (regex) for each interface searches for two lines, one that includes the `line protocol is` text and one that includes the *Input queue* text:

```

router#show interfaces | include line protocol is|Input queue
GigabitEthernet0/0 is up, line protocol is up
  Input queue: 101/4000/60902/0 (size/max/drops/flushes); Total output drops: 0
GigabitEthernet0/1 is up, line protocol is up
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
GigabitEthernet0/2 is up, line protocol is up

```

```

    Input queue: 0/2000/0/0 (size/max/drops/flushes); Total output drops: 0
GigabitEthernet0/2.403 is up, line protocol is up
Loopback0 is up, line protocol is up
    Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
Loopback1 is up, line protocol is up
    Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0

```

To avoid subinterfaces, the regex will only match on interface names that do not contain a period character, as in GigabitEthernet0/2.403. To match the interface name, the following regex will be used:

```

^[^\.\s]+ is .*, line protocol is

```

In the above example, the `^` character matches the beginning of the line. The `[^\.\s]+` character sequence matches the IOS interface name (type and number), which is any sequence of characters that does not contain a period character or whitespace (space, tab, new line or carriage return). The `is .* line protocol is` sequence matches the line interface state, and "line protocol" is text.

An example at the IOS command prompt follows:

```

Router#show interfaces | include ^[^\.\s]+ is .*, line protocol is
GigabitEthernet0/0 is up, line protocol is up
GigabitEthernet0/1 is up, line protocol is up
GigabitEthernet0/2 is up, line protocol is up
Loopback0 is up, line protocol is up

```

Note that the regex repetition qualifiers "one or more" (+) and "zero or more" (*) in IOS match the minimum characters. In regular expression terminology, these qualifiers are *non-greedy*. In contrast, a question mark character (?) must follow the Tcl regex repetition modifiers to match on the fewest characters. This adjustment modifies the regex to the following:

```

^[^\.\s]+ is .*?, line protocol is

```

The Tcl **regex** command will also store several sections of text; these sections are referred to as variables and will be noted with parentheses before and after the regex that matches the variables:

- Interface name
- Line protocol state
- Current input queue size
- Maximum hold queue size

The final regex pattern enclosed in curly brackets that is required by the **regex** Tcl command is as follows:

```

{^[^\.\s]+) is .*?, line protocol is (\S+).*?\s*Input queue: (\d+)/(\d+)/}

```

In the above example, the `^[^\.\s]+)` character sequence matches the interface name, excluding subinterfaces, and saves the interface name in a variable. The `is .*?, line protocol is` sequence matches the interface state, and the ", line protocol is " sequence is text. The `(\S+)` character sequence matches and stores the line protocol state (non-whitespace) in a variable. The `.*?\s*Input queue:` sequence

matches all text before and including the space after a new line followed by whitespace and the "Input queue": text. This portion of the regex can cross multiple lines of **show interfaces** command output and will require the **-lineanchor** option to the **regex** Tcl command. The **(\d+)/(\d+)/** character sequence matches and stores a variable composed of one or more decimal digits followed by a **/** character, another variable composed of decimal digits, and then another **/** character. These variables represent the current and maximum hold queue size.

To verify the operation of the Tcl script, an EEM variable will be used to help debug the script. The variable is **EEM_INTERFACE_INPUT_Q_DEBUG**, and the script will send a syslog message for each interface that it finds with the relevant input hold queue information. The variable must be set to **yes** (case ignored) for the syslog messages to be sent. A sample message with the **EEM_INTERFACE_INPUT_Q_DEBUG** variable set to *Yes* follows:

```
%HA_EM-7-LOG: system:/lib/tcl/eem_scripts_registered/interface-input-q.tcl:
Interface GigabitEthernet0/0 input queue at 101 of 4000
```

The Tcl script can be copied to the vulnerable device, and if the IOS file system supports directories, a directory for EEM policies will be created. The following example is a copy of the script to the **disk0:** directory **EEM** file system:

```
router#mkdir DISK0:/EEM
Create directory filename [EEM]?
Created dir disk0:/EEM
router#copy <location of tcl script><location of Tcl script> disk0:/EEM/interf
</pre>
```

Then the Tcl script can be registered as an EEM policy with the following commands:

```
!-- Define variable verify operation of policy
event manager environment EEM_INTERFACE_INPUT_Q_DEBUG No

!-- Location where the tcl scripts will be found
event manager directory user policy disk0:/EEM

!-- register the script as an EEM policy
event manager policy interface-input-q.tcl type user
```

Once registered, the EEM policy will send a syslog message when an interface hold queue exceeds its maximum length. An example syslog message follows:

```
%HA_EM-7-LOG: system:/lib/tcl/eem_scripts_registered/interface-input-q.tcl: Int
Input queue: 4001/4000 (size/max)
```

While an EEM policy is being tested, the **EEM_INTERFACE_INPUT_Q_DEBUG** variable can be used to instruct the policy to send a syslog message for each interface and each interval, regardless of its queue length.

The TCL script is available for [download](#) at the [Cisco Beyond: Embedded Event Manager \(EEM\) Scripting Community](#) and is reproduced as follows:

```

#
# This EEM policy detects full interface input queues. When a full queue
# has been detected, a syslog message is sent. The number of items in each
# interface input queue, as well as the maximum queue length, is determined
# using the EXEC command 'show interfaces'.
#
# The following is an example of the syslog message produced by this policy.
#
# %HA_EM-4-LOG: tmpsys:/eem_policy/interface-input-q.tcl: Interface
# Ethernet0/0 input queue full. Input queue: 76/75 (size/max)
#
# v20080304
#

#
# Register this policy as time based using a watchdog timer. It will be
# triggered every 60 seconds.
#
::cisco::eem::event_register_timer watchdog name sixtySeconds time 60

# Import the cisco namespaces that allows access to CLI procedures.
namespace import ::cisco::eem::*

#
# The environment variable EEM_INTERFACE_INPUT_Q_DEBUG must exist for this
# script to run. If this variable is set to 'yes', the queue depth of all
# interfaces is reported via syslog. Otherwise, syslog messages will only
# be sent for queues where the queue depth exceeds the configured maximum.
#
if {[info exists EEM_INTERFACE_INPUT_Q_DEBUG]} {
    set result \
    "Policy cannot be run: variable EEM_INTERFACE_INPUT_Q_DEBUG has not been set"
    error $result $errorInfo
}

#
# Obtain the output of the 'show interfaces' command and place it into the
# variable cmd_output.
#
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli $result
}
if [catch {cli_exec $cli(fd) "show interfaces"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_close $cli(fd) $cli(tty_id)} result] {
    error $result $errorInfo
}

#
# Parse the command output using a regular expression and place the results
# in the list variable named list.
#
# Subinterfaces will be skipped by matching only interface names that do not
# contain a '.'.
#

```

```

catch { \
  regexp -all -inline -lineanchor \
  {^([\^.\s]+?) is .*?, line protocol is (\S+).*?\s*Input queue: (\d+)/(\d+)/}
  $cmd_output
} list

#
# After cmd_output has been parsed by the regular expression, the variable
# list contains a set of data for each interface. Each set contains five
# items, four of which were specifically stored using ()s within the regexp
# command. The five items are:
#
# - the entire matched string
# - interface name
# - interface line protocol
# - input queue size
# - input queue max
#
set list_index 0
set stored_items 4

#
# Perform the following code for each interface in the list variable list.
#
while { $list_index < [llength $list] } {

  # Copy the data from the list to more readable variable names
  set interface_name      [lindex $list [expr $list_index + 1]]
  set line_protocol_state [lindex $list [expr $list_index + 2]]
  set input_queue_size    [lindex $list [expr $list_index + 3]]
  set input_queue_max     [lindex $list [expr $list_index + 4]]

  #
  # Send a syslog message if an interface is up, and its queue depth exceeds
  # the input queue size.
  #
  if { $line_protocol_state == "up"
      && ($input_queue_size >= $input_queue_max) } {

    action_syslog priority warning msg \
      "Interface $interface_name input queue full.\
      Input queue: $input_queue_size/$input_queue_max (size/max)"

  } else {
    if { [string compare -nocase $EEM_INTERFACE_INPUT_Q_DEBUG "yes"] == 0 } {

      #
      # If debugging has been enabled via the environment variable, send a syslog
      # for each interface regardless of queue depth.
      #
      action_syslog priority debug msg \
        "Interface $interface_name input queue at $input_queue_size of $input_queue

    }
  }

  # Advance the list_index to the next set of items in the list
  incr list_index [expr $stored_items + 1]
}

```

Cisco IOS NetFlow

Identification: Traffic Flow Identification Using IPv6 NetFlow Records

Administrators can configure Cisco IOS IPv6 NetFlow on Cisco IOS routers and switches to aid in the identification of traffic flows that may be attempts to exploit the vulnerability described in this document. Administrators are advised to investigate flows to determine whether they are attempts to exploit the vulnerability or whether they are legitimate traffic flows.

A sample IPv6 Netflow configuration follows; note that NetFlow has been configured to export records to a workstation with the IPv4 address of 192.168.132.91 using UDP port 10000:

```
ipv6 flow-export destination 192.168.132.91 10000
ipv6 flow-capture packet-length
ipv6 flow-capture ttl
ipv6 flow-capture vlan-id
ipv6 flow-capture icmp
ipv6 flow-capture ip-id
ipv6 flow-capture mac-addresses

!

interface GigabitEthernet0/0
 ip address 192.168.208.20 255.255.255.0
 ipv6 address 2001:DB8:1:208::20/64
 ipv6 flow ingress
```

Administrators should investigate flows to determine whether they are attempts to exploit this vulnerability or whether they are legitimate traffic flows.

```
router#show ipv6 flow cache
IP packet size distribution (50078919 total packets):
  1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
  .000 .990 .001 .008 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

      512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
      .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 475168 bytes
  8 active, 4088 inactive, 6160 added
  1092984 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes

  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 33928 bytes
  16 active, 1008 inactive, 12320 added, 6160 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
SrcAddress      InpIf      DstAddress      OutIf      Prot  SrcPrt  DstPrt  Packets
2001:DB...06::201 Gi0/0    2001:DB...28::20 Local    0x11 0x13C4 0x0035 1464K
2001:DB...6A:5BA6 Gi0/0      2001:DB...28::21 Gi0/1      0x3A 0x0000 0x8000 1191
2001:DB...6A:5BA6 Gi0/0      2001:DB...134::3 Gi0/1      0x3A 0x0000 0x8000 1191
2001:DB...6A:5BA6 Gi0/0      2001:DB...128::4 Gi0/1      0x3A 0x0000 0x8000 1192
2001:DB...6A:5BA6 Gi0/0    2001:DB...128::2 Gi0/1    0x11 0x160A 0x0035 1597
2001:DB...6A:5BA6 Gi0/0      2001:DB...128::3 Gi0/1      0x3A 0x0000 0x8000 1192
2001:DB...6A:5BA6 Gi0/0      2001:DB...146::3 Gi0/1      0x3A 0x0000 0x8000 1192
```

```
2001:DB...6A:5BA6 Gi0/0      2001:DB...144::4 Gi0/1      0x3A 0x0000 0x8000 1193
```

Note: To permit display of the full 128 bit IPv6 address, please use the **terminal width 132** exec mode command.

In the preceding example, there are multiple IPv6 flows for DNS packets on UDP port 53 (**hex value 0x0035**). The UDP packets in these flows may be spoofed and may indicate an attempt to exploit the vulnerability described in this document. Administrators should compare these flows to baseline utilization for TACACS packets on UDP port 49, DNS packets on UDP port 53, RSVP packets on UDP port 1698, L2TP/L2F packets on UDP port 1701, IP SLA packets on UDP port 1967, MGCP packets on UDP port 2427, and SIP packets on UDP port 5060 and also investigate the flows to determine whether they are sourced from untrusted hosts or networks.

To view only the traffic flows for IPv6 DNS packets on UDP port 53 (**hex value 0035**), the command **show ipv6 flow cache | include SrcAddress|_0x11_.*0x0035** will display the related NetFlow records as shown here:

```
router#show ipv6 flow cache | include SrcAddress|_0x11_.*0x0035
SrcAddress      InpIf  DstAddress      OutIf  Prot  SrcPrt  DstPrt  Packets
2001:DB...06::201 Gi0/0    2001:DB...28::20 Local   0x11  0x189C  0x0035  56K
2001:DB...06::201 Gi0/0    2001:DB...28::20 Local   0x11  0x18B0  0x0035  74
2001:DB...06::201 Gi0/0    2001:DB...28::20 Local   0x11  0x18D8  0x0035  356
router#
```

To view the traffic flows for **TACACS** packets on **UDP port 49 (Hex value 0031)**, **DNS** packets on **UDP port 53 (Hex value 0035)**, **RSVP** packets on **UDP port 1698 (Hex value 06A2)**, **L2TP/L2F** packets on **UDP port 1701 (Hex value 06A5)**, **IP SLA** packets on **UDP port 1967 (Hex value 07AF)**, **MGCP** packets on **UDP port 2427 (Hex value 097B)**, and **SIP** packets on **UDP port 5060 (Hex value 013C4)**, the command **show ip cache flow | include SrcIf|_0x11_.*0x(0031|0035|06A2|06A5|0031|0035|06A2|06A5|07AF|097B|13C4)** will display the related NetFlow records as shown here:

UDP Flows

```
router#show ipv6 cache flow | include SrcAddress|_0x11_.*0x(0031|0035|06A2|06A5
SrcAddress      InpIf  DstAddress      OutIf  Prot  SrcPrt  DstPrt  Packets
2001:DB...06::220 Gi0/0    2001:DB...144::2 Gi0/1    0x11  0x17FC  0x097B  3853
2001:DB...06::220 Gi0/0    2001:DB...144::2 Gi0/1    0x11  0x17E8  0x097B  4806
2001:DB...06::230 Gi0/0    2001:DB...34::21 Gi0/1    0x11  0x1798  0x13C4  21
2001:DB...06::230 Gi0/0    2001:DB...34::21 Gi0/1    0x11  0x1770  0x13C4  357
2001:DB...06::201 Gi0/0    2001:DB...28::20 Local    0x11  0x189C  0x0035  47
2001:DB...06::201 Gi0/0    2001:DB...28::20 Local    0x11  0x18C4  0x0035  980
2001:DB...06::201 Gi0/0    2001:DB...28::20 Local    0x11  0x18EC  0x0035  11K
2001:DB...06::220 Gi0/0    2001:DB...144::2 Gi0/1    0x11  0x1810  0x097B  5500
router#
```

Applications that process NetFlow Records such as [Cisco NetFlow Collector](#) can also be used to visualize this information, which is illustrated in the following screen shot:

Report: Custom - 2007-11-29 12:46:23.733
29 Nov 2007 11:44:00 - 29 Nov 2007 12:44:00

	Device	protocol	srcaddr	dstaddr	pkts	octets
1.	*	domain	2001:db8:1:208:0:0:0:201	2001:db8:1:128:0:0:0:20	133988	6431424
2.	*	OTHER	2001:db8:1:208:214:5eff:fe6a:5ba6	fe80:0:0:0:218:74ff:feb5:a41b	6	384
3.	*	OTHER	fe80:0:0:0:203:32ff:fe2e:3400	ff02:0:0:0:0:0:0:1	2	208

In the preceding example, Cisco NetFlow Collector shows **133988** DNS packets on UDP Port 53 with destination of 2001:db8:1:208::20.

Cisco ASA, PIX, and FWSM Firewalls

Mitigation: IPv6 Transit Access Control Lists

To protect the network from traffic that enters the network at ingress access points, which may include Internet connection points, partner and supplier connection points, or VPN connection points, administrators are advised to deploy tACLs to perform policy enforcement. Administrators can construct a tACL by explicitly permitting only authorized traffic to enter the network at ingress access points or permitting authorized traffic to transit the network in accordance with existing security policies and configurations. A tACL workaround cannot provide complete protection against the vulnerability that has a network attack vector when the attack comes from a trusted source address.

The IPv6 tACL policy denies unauthorized TACACS packets on UDP port 49, DNS packets on UDP port 53, RSVP packets on UDP port 1698, L2TP/L2F packets on UDP port 1701, IP SLA packets on UDP port 1967, MGCP packets on UDP port 2427, and SIP packets on UDP port 5060 that are sent to affected devices. In the following example, 2001:db8:1:128::/64 is the IPv6 address space that is used by the affected devices. To mitigate against exploits of this vulnerability, all IPv6 traffic on vulnerable ports that is addressed to the affected devices must be denied. Care should be taken to allow required traffic for routing and administrative access prior to denying all unauthorized traffic.

Additional information about tACLs is available in [Transit Access Control Lists: Filtering at Your Edge](#).

```

!-- This vulnerability does not allow trusted
!-- sources access over IPv6 on the vulnerable ports

!-- The following vulnerability-specific access control entries
!-- (ACEs) can aid in identification of attacks

ipv6 access-list IPv6-Transit-ACL-Policy deny udp any 2001:db8:1:128::/64 eq ta
ipv6 access-list IPv6-Transit-ACL-Policy deny udp any 2001:db8:1:128::/64 eq do
ipv6 access-list IPv6-Transit-ACL-Policy deny udp any 2001:db8:1:128::/64 eq 16
ipv6 access-list IPv6-Transit-ACL-Policy deny udp any 2001:db8:1:128::/64 eq 17
ipv6 access-list IPv6-Transit-ACL-Policy deny udp any 2001:db8:1:128::/64 eq 19
ipv6 access-list IPv6-Transit-ACL-Policy deny udp any 2001:db8:1:128::/64 eq 24
ipv6 access-list IPv6-Transit-ACL-Policy deny udp any 2001:db8:1:128::/64 eq si

!-- Permit/deny all other Layer 3 and Layer 4 traffic in accordance
!-- with existing security policies and configurations

!-- Explicit deny for all other IP traffic

ipv6 access-list IPv6-Transit-ACL-Policy deny ip any any

!-- Apply tACL to interfaces in the ingress direction

access-group IPv6-Transit-ACL-Policy in interface outside

```

Mitigation: Spoofing Protection Using Unicast Reverse Path Forwarding

This vulnerability can be exploited by spoofed IP packets. Administrators can deploy and configure Unicast RPF as a protection mechanism against spoofing.

Unicast RPF is configured at the interface level and can detect and drop packets that lack a verifiable source IP address. Administrators should not rely on Unicast RPF to provide 100 percent spoofing protection because spoofed packets may enter the network through a Unicast RPF-enabled interface if an appropriate return route to the source IP address exists. In an enterprise environment, Unicast RPF might be enabled at the Internet edge and at the internal access layer on the user-supporting Layer 3 interfaces.

For additional information about the configuration and use of Unicast RPF, reference the Cisco Security Appliance Command Reference for [ip verify reverse-path](#) and the [Understanding Unicast Reverse Path Forwarding](#) Applied Intelligence white paper.

Identification: IPv6 Transit Access Control Lists

After the tACL has been applied to an interface, administrators can use the **show ipv6 access-list** command to identify the number of TACACS packets on UDP port 49, DNS packets on UDP port 53, RSVP packets on UDP port 1698, L2TP/L2F packets on UDP port 1701, IP SLA packets on UDP port 1967, MGCP packets on UDP port 2427, and SIP packets on UDP port 5060 that have been filtered. Administrators should investigate filtered packets to determine whether they are attempts to exploit this vulnerability. Example output for **show access-list IPv6-Transit-ACL-Policy** follows:

```

firewall#show ipv6 access-list IPv6-Transit-ACL-Policy
ipv6 access-list IPv6-Transit-ACL-Policy; 8 elements
ipv6 access-list IPv6-Transit-ACL-Policy line 1 deny udp any 2001:db8:1:128::/6

```

```
ipv6 access-list IPv6-Transit-ACL-Policy line 2 deny udp any 2001:db8:1:128::/6
ipv6 access-list IPv6-Transit-ACL-Policy line 3 deny udp any 2001:db8:1:128::/6
ipv6 access-list IPv6-Transit-ACL-Policy line 4 deny udp any 2001:db8:1:128::/6
ipv6 access-list IPv6-Transit-ACL-Policy line 5 deny udp any 2001:db8:1:128::/6
ipv6 access-list IPv6-Transit-ACL-Policy line 6 deny udp any 2001:db8:1:128::/6
ipv6 access-list IPv6-Transit-ACL-Policy line 7 deny udp any 2001:db8:1:128::/6
ipv6 access-list IPv6-Transit-ACL-Policy line 8 deny ip any any (hitcnt=0)
firewall#
```

In the preceding example, access list *IPv6-Transit-ACL-Policy* has dropped the following packets received from an untrusted host or network:

- **1690 IPv6 DNS** packets on **UDP** port **53**.
- **501 IPv6 IP SLA** packets on **UDP** port **1967**.

Identification: Firewall Access-list Syslog Messages

Firewall syslog message *106023* will be generated for packets denied by an access control entry (ACE) that does not have the **log** keyword present. Additional information about this syslog message is available in [Cisco Security Appliance System Log Message - 106023](#).

Information about configuring syslog for the Cisco ASA 5500 Series Adaptive Security Appliance or the Cisco PIX 500 Series Security Appliance is available in [Configuring Logging on the Cisco Security Appliance](#). Information about configuring syslog on the FWSM for Cisco Catalyst 6500 Series switches and Cisco 7600 Series routers is available in [Configuring Monitoring and Logging on the Cisco FWSM](#).

In the following example, the **show logging | grep regex** command extracts syslog messages from the logging buffer on the firewall. These messages provide additional information about denied packets that could indicate attempts to exploit the vulnerability described in this document. It is possible to use different regular expressions with the **grep** keyword to search for specific data in the logged messages.

Additional information about regular expression syntax is available in [Using the Command Line Interface](#).

```
firewall#show logging | grep 106023
Dec 03 2007 12:09:05: %ASA-4-106023: Deny udp src outside:2001:db8:1:206::201/6
dst inside:2001:db8:1:128::2/53 by access-group "IPv6-Transit-ACL-Policy"
Dec 03 2007 12:13:35: %ASA-4-106023: Deny udp src outside:2001:db8:1:206::220/6
dst inside:2001:db8:1:144::2/53 by access-group "IPv6-Transit-ACL-Policy"
firewall#
```

In the preceding example, the messages logged for the tACL *IPv6-Transit-ACL-Policy* show potentially spoofed IPv6 **DNS** packets on **UDP** port **53** sent to the address block assigned to the infrastructure devices

Additional information about syslog messages for ASA and PIX security appliances is available in [Cisco Security Appliance System Log Messages](#). Additional information about syslog messages for the FWSM is available in [Catalyst 6500 Series Switch and Cisco 7600 Series Router Firewall Services Module Logging Configuration and System Log Messages](#).

Identification: Spoofing Protection Using Unicast Reverse Path Forwarding

Firewall syslog message *106021* will be generated for packets denied by Unicast RPF. Additional information about this syslog message is available in [Cisco Security Appliance System Log Message - 106021](#).

Information about configuring syslog for the Cisco ASA 5500 Series Adaptive Security Appliance or the Cisco PIX 500 Series Security Appliance is available in [Configuring Logging on the Cisco Security Appliance](#). Information about configuring syslog on the FWSM for Cisco Catalyst 6500 Series switches and Cisco 7600 Series routers is available in [Configuring Monitoring and Logging on the Cisco FWSM](#).

In the following example, the **show logging | grep regex** command extracts syslog messages from the logging buffer on the firewall. These messages provide additional information about denied packets that could indicate attempts to exploit the vulnerability described in this document. It is possible to use different regular expressions with the **grep** keyword to search for specific data in the logged messages.

Additional information about regular expression syntax is available in [Using the Command Line Interface](#).

```
firewall#show logging | grep 106021
%ASA-1-106021: Deny UDP reverse path check from 2001:db8:1:200::201
to 2001:db8:1:128::2 on interface outside
%ASA-1-106021: Deny UDP reverse path check from 2001:db8:1:200::201
to 2001:db8:1:128::2 on interface outside
firewall#
```

The **show asp drop** command can also identify the number of packets that Unicast RPF has dropped, as shown in the following example:

```
firewall#show asp drop

Frame drop:

  Reverse-path verify failed          381100
  Flow is denied by configured rule    855
  Expired flow                        1
  Interface is down                   2

Flow drop:

firewall#
```

In the preceding example, Unicast RPF has dropped **381100 packets** received on interfaces with Unicast RPF configured.

For additional information about the configuration and use of IPv6 Unicast RPF, reference the Cisco Security Appliance Command Reference for [show asp drop](#).

Additional Information

THIS DOCUMENT IS PROVIDED ON AN "AS IS" BASIS AND DOES NOT IMPLY ANY KIND OF GUARANTEE OR WARRANTY, INCLUDING THE WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR USE. YOUR USE OF THE INFORMATION ON THE DOCUMENT OR MATERIALS LINKED FROM THE DOCUMENT IS AT YOUR OWN RISK.

CISCO RESERVES THE RIGHT TO CHANGE OR UPDATE THIS DOCUMENT AT ANY TIME.

Revision History

Revision 1.1	2008- April-14	Include local link addresses in IOS tACL and match ND permits
Revision 1.0	2008- March-26	Initial public release

Cisco Security Procedures

Complete information on reporting security vulnerabilities in Cisco products, obtaining assistance with security incidents, and registering to receive security information from Cisco, is available on Cisco's worldwide website at

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.html. This includes instructions for press inquiries regarding Cisco security notices. All Cisco security advisories are available at <http://www.cisco.com/go/psirt>.

Related Information

- [Cisco Applied Mitigation Bulletins](#)
- [Protecting Your Core: Infrastructure Protection Access Control Lists](#)
- [Transit Access Control Lists: Filtering at Your Edge](#)
- [Understanding Access Control List Logging](#)
- [A Security-Oriented Approach to IP Addressing](#)
- [Identifying Incidents Using Firewall and IOS Router Syslog Events](#)
- [Understanding Unicast Reverse Path Forwarding](#)
- [Cisco IOS IPv6 Command Reference](#)
- [Unicast RPF for IPv6 on the Cisco 12000 Series](#)
- [Cisco IOS NetFlow - Home Page on Cisco.com](#)
- [Cisco IOS NetFlow White Papers](#)
- [Writing Embedded Event Manager Policies Using Tcl](#)
- [Cisco Beyond: Embedded Event Manager \(EEM\) Scripting Community](#)
- [Cisco Network Foundation Protection White Papers](#)
- [Cisco Network Foundation Protection Presentations](#)
- [Cisco Firewall Products - Home Page on Cisco.com](#)
- [Unicast Reverse Path Forwarding Loose Mode](#)
- [Unicast Reverse Path Forwarding Enhancements for the Internet Service Provider - Internet Service Provider Network Edge](#)
- [Cisco Security Monitoring, Analysis, and Response System](#)
- [Common Vulnerabilities and Exposures \(CVE\)](#)

Help us help you.



Please rate this document.

- Excellent
- Good
- Average
- Fair
- Poor

This document solved my problem.

- Yes
- No
- Just browsing

Suggestions for improvement:

(256 character limit)

Home	How to Buy	Login	Profile	Feedback	Site Map	Help
----------------------	----------------------------	-----------------------	-------------------------	--------------------------	--------------------------	----------------------

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2007 - 2008 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)