

# Cisco Applied Mitigation Bulletin: Identifying and Mitigating Exploitation of the Cisco IOS Software Session Initiation Protocol and Crafted UDP Vulnerabilities

<http://www.cisco.com/warp/public/707/cisco-amb-20090325-sip-and-udp.shtml>

## Revision 1.0

For Public Release 2009 March 25 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 following PSIRT Security Advisories: *Cisco IOS Software Session Initiation Protocol Denial of Service Vulnerability* and *Cisco IOS Software Multiple Features Crafted UDP Packet Vulnerabilities* and provides identification and mitigation techniques that administrators can deploy on Cisco network devices.

## Vulnerability Characteristics

There are multiple vulnerabilities in Cisco IOS Software. The following subsections summarize these vulnerabilities:

**Cisco IOS Software Session Initiation Protocol Denial of Service Vulnerability:** This vulnerability can be exploited remotely without authentication and without end-user interaction. Successful exploitation of this vulnerability may cause the affected device to crash. Repeated attempts to exploit this vulnerability could result in a sustained DoS condition. The attack vector for exploitation is through a Session Initiation Protocol (SIP) packet using UDP port 5060, TCP port

5060, or TCP port 5061. An attacker could exploit this vulnerability using spoofed packets.

This vulnerability has been assigned CVE identifier CVE-2009-0636.

**Cisco IOS Software Multiple Features Crafted UDP Packet Vulnerability:** This vulnerability can be exploited remotely without authentication and without end-user interaction. Successful exploitation of this vulnerability may result in a denial of service (DoS) condition. Repeated attempts to exploit this vulnerability could result in a sustained DoS condition. An attacker could exploit this vulnerability using spoofed packets.

The attack vectors for exploitation are through packets using the following protocols and ports:

- IP Service Level Agreement (SLA) Responder using UDP port 1967
- SIP using UDP port 5060
- H.323 Call Signaling Transport using UDP port 2517
- Media Gateway Control Protocol (MGCP) using UDP port 2427

This vulnerability has been assigned CVE identifier CVE-2009-0631.

Information about vulnerable, unaffected, and fixed software is available in the respective PSIRT Security Advisories, which are available at the following links:

Cisco IOS Session Initiation Protocol Denial of Service Vulnerability:

- <http://www.cisco.com/warp/public/707/cisco-sa-20090325-sip.shtml>

Crafted UDP packet affects multiple Cisco IOS features:

- <http://www.cisco.com/warp/public/707/cisco-sa-20090325-udp.shtml>

## Mitigation Technique Overview

Cisco devices provide several countermeasures for these vulnerabilities. Administrators are advised to consider these protection methods to be general security best practices for infrastructure devices and the traffic that transits the network. This section of the document provides an overview of these techniques.

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

- Infrastructure access control lists (iACLs)
- Unicast Reverse Path Forwarding (Unicast RPF)
- IP source guard (IPSG)

These protection mechanisms filter and drop, as well as verify the source IP address of, packets that are attempting to exploit these vulnerabilities.

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

The proper deployment and configuration of IPSG provides an effective means of protection against spoofing attacks at the access layer.

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:

- Transit access control lists (tACLs)
- Unicast Reverse Path Forwarding (Unicast RPF)

These protection mechanisms filter and drop, as well as verify the source IP address of, packets that are attempting to exploit these vulnerabilities.

Effective use of Cisco Embedded Event Manager (EEM) policies provides visibility into exploit attempts by providing administrators with a variety of identification options, including monitoring counters, sending syslog messages, and sending SNMP traps

Cisco IOS NetFlow flow records can provide visibility into network-based exploitation attempts.

Cisco IOS Software, Cisco ASA appliances, 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 are advised to follow their standard risk evaluation and mitigation processes to determine the potential impact of these vulnerabilities. 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](#) can help organizations develop repeatable security evaluation and response processes.

## Device-Specific Mitigation and Identification



**Caution:** The effectiveness of any mitigation technique depends 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: 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. An iACL workaround cannot provide complete protection against these vulnerabilities when the attack originates from a trusted source address.

The iACL policy denies unauthorized packets that are sent to affected devices via the following protocols:

- SIP using TCP port 5060
- SIP using TCP port 5061
- SIP using UDP port 5060
- IP SLA Responder using UDP port 1967
- H.323 Call Signaling Transport using UDP port 2517
- MGCP using UDP port 2427

In the following example, 192.168.60.0/24 is the IP address space that is used by the affected devices, and the host at 192.168.100.1 is considered a trusted source that requires access to the affected devices. Care should be taken to allow required traffic for routing and administrative access prior to denying all unauthorized traffic. Whenever possible, infrastructure address space should be distinct from the address space used for user and services segments. Using this addressing methodology will assist with the construction and deployment of iACLs.

Additional information about iACLs is in [Protecting Your Core: Infrastructure Protection Access Control Lists](#).

```
ip access-list extended Infrastructure-ACL-Policy

!
!-- Include explicit permit statements for trusted
!-- sources that require access on the vulnerable ports
!

permit tcp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 5060
permit tcp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 5061
permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 5060
permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 1967
permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 2517
permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 2427

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

deny tcp any 192.168.60.0 0.0.0.255 eq 5060
deny tcp any 192.168.60.0 0.0.0.255 eq 5061
deny udp any 192.168.60.0 0.0.0.255 eq 5060
deny udp any 192.168.60.0 0.0.0.255 eq 1967
deny udp any 192.168.60.0 0.0.0.255 eq 2517
deny udp any 192.168.60.0 0.0.0.255 eq 2427

!
!-- Explicit deny ACE for traffic sent to addresses configured within
!-- the infrastructure address space
!

deny ip any 192.168.60.0 0.0.0.255

!
!-- 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
 ip access-group 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 ip unreachables**. ICMP unreachable rate limiting can be changed from the default using the global configuration command **ip icmp rate-limit unreachable *interval-in-ms***.

## Mitigation: Spoofing Protection

### Unicast Reverse Path Forwarding

The vulnerabilities that are described in this document can be exploited by spoofed IP packets. Administrators can deploy and configure Unicast Reverse Path Forwarding (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 complete 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. Administrators are advised to take care to ensure that the appropriate 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, Unicast RPF might be enabled at the Internet edge and the internal access layer on the user-supporting Layer 3 interfaces.

Additional information is in the [Unicast Reverse Path Forwarding Loose Mode Feature Guide](#).

For additional information about the configuration and use of Unicast RPF, reference the [Understanding Unicast Reverse Path Forwarding](#) Applied Intelligence white paper.

### IP Source Guard

IP source guard (IPSG) is a security feature that restricts IP traffic on nonrouted, Layer 2 interfaces by filtering packets based on the DHCP snooping binding database and manually configured IP source bindings. Administrators can use IPSG to prevent attacks from an attacker who attempts to spoof packets by forging the source IP address and/or the MAC address. When properly deployed and configured, IPSG coupled with strict mode Unicast RPF provides the most effective means of spoofing protection for the vulnerabilities that are described in this document.

Additional information about the deployment and configuration of IPSG is in [Configuring DHCP Features and IP Source Guard](#).

### Identification: Infrastructure Access Control Lists

After the administrator applies the iACL to an interface, the **show ip access-lists** command will identify the number of higher layer protocol packets on their respective ports that have been filtered on interfaces which the iACL is applied. Administrators should investigate filtered packets to determine whether they are attempts to exploit these vulnerabilities. Example output for **show ip access-lists Infrastructure-ACL-Policy** follows:

```
router#show ip access-lists Infrastructure-ACL-Policy
```

```

Extended IP access list Infrastructure-ACL-Policy
 10 permit tcp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 5060 (21 matches)
 20 permit tcp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 5061 (63 matches)
 30 permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 5060 (17 matches)
 40 permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 1967 (31 matches)
 50 permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 2517 (11 matches)
 60 permit udp host 192.168.100.1 192.168.60.0 0.0.0.255 eq 2427 (16 matches)
 70 deny tcp any 192.168.60.0 0.0.0.255 eq 5060 (13 matches)
 80 deny tcp any 192.168.60.0 0.0.0.255 eq 5061 (17 matches)
 90 deny udp any 192.168.60.0 0.0.0.255 eq 5060 (36 matches)
100 deny udp any 192.168.60.0 0.0.0.255 eq 1967 (10 matches)
110 deny udp any 192.168.60.0 0.0.0.255 eq 2517 (9 matches)
120 deny udp any 192.168.60.0 0.0.0.255 eq 2427 (21 matches)
130 deny ip any 192.168.60.0 0.0.0.255 (127 matches)
router#

```

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

- **13 SIP** packets on **TCP port 5060** for ACE line 70
- **17 SIP** packets on **TCP port 5061** for ACE line 80
- **36 SIP** packets on **UDP port 5060** for ACE line 90
- **10 IP SLA Responder** packets on **UDP port 1967** for ACE line 100
- **9 H.323 Call Signaling Transport** packets on **UDP port 2517** for ACE line 110
- **21 MGCP** packets on **UDP port 2427** for ACE line 120

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** and **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 Cisco IOS Software, the **ip access-list logging interval** *interval-in-ms* command can limit the effects of process switching induced by ACL logging. The **logging rate-limit** *rate-per-second* [**except** *loglevel*] command limits the impact of log generation and transmission.

The CPU impact from ACL logging can be addressed in hardware on the Cisco Catalyst 6500 Series switches and Cisco 7600 Series routers with Supervisor Engine 720 or Supervisor Engine 32 using optimized ACL logging.

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 Unicast Reverse Path Forwarding

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

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

```
router#show cef interface GigabitEthernet 0/0 internal | include drop
--          CLI Output Truncated          --
  ip verify: via=rx (allow default), acl=0, drop=18, sdrop=0
router#
```

**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 ip interface GigabitEthernet 0/0 | begin verify
--          CLI Output Truncated          --
  IP verify source reachable-via RX, allow default, allow self-ping
  11 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    27            0           0           18       0       0
router#
```

```
router#show ip traffic

IP statistics:
Rcvd:  68051015 total, 2397325 local destination
       43999 format errors, 0 checksum errors, 33 bad hop count
       2 unknown protocol, 929 not a gateway
       21 security failures, 190123 bad options, 542768 with options
Opts:  352227 end, 452 nop, 36 basic security, 1 loose source route
       45 timestamp, 59 extended security, 41 record route
       53 stream ID, 3 strict source route, 40 alert, 45 cipso, 0 ump
       361634 other
Frag:  0 reassembled, 10008 timeouts, 56866 couldn't reassemble
       0 fragmented, 0 fragments, 0 couldn't fragment
Bcast: 64666 received, 0 sent
Mcast: 1589885 received, 2405454 sent
Sent:  3001564 generated, 65359134 forwarded
Drop:  4256 encapsulation failed, 0 unresolved, 0 no adjacency
       18 no route, 18 unicast RPF, 0 forced drop
       0 options denied
Drop:  0 packets with source IP address zero
Drop:  0 packets with internal loop back IP address
--          CLI Output Truncated          --
router#
```

In the preceding **show cef drop** and **show ip traffic** examples, Unicast RPF has dropped **18 IP packets** received globally on all interfaces with Unicast RPF configured because of the inability to verify the source address of the IP packets within the Forwarding Information Base of Cisco Express Forwarding.

## Identification: Embedded Event Manager

A Tcl-based Embedded Event Manager (EEM) policy can be used on a vulnerable Cisco IOS device to detect the interface input blockage ("wedge") condition that is caused by these vulnerabilities. This policy allows administrators to monitor the hold queue counter directly for all interfaces of the vulnerable device. When EEM detects potential exploitation of these vulnerabilities, the EEM policy can trigger a response by sending a syslog message or an 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 produces a syslog message only when the hold queue of any interface in the device reaches its maximum configured value.

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

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

**Tcl Regular Expression Syntax** To avoid subinterfaces, the regex will match interface names that do not contain a period character. To match the interface name, the following regex will be used:

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

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

The following example illustrates using the **show interfaces** command with a regex at the Cisco IOS device command prompt:

```
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
```

The regex repetition qualifiers *one or more* (+) and *zero or more* (\*) in the Cisco IOS command line interface match the minimum characters. In regular expression terminology, these qualifiers are "non-greedy." In contrast, within Tcl a question mark character (?) must follow the regex repetition qualifiers when the intention is to match the fewest characters. This adjustment modifies the regex to the following:

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

The following example illustrates the final regex pattern, which is enclosed in curly brackets as required by the Tcl regex:

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

The Tcl **regexp** command will also store several sections of text. These sections are referred to as *variables*. Each portion of the regex that matches these variables is enclosed in parentheses. The following variables are stored using the regex:

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

In the preceding 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. The `(\S+)` character sequence matches and stores the line protocol state in a variable. The `.*?\s*Input queue:` sequence matches all text before and including the space after a new line followed by white space and the **Input queue:** text. This portion of the regex can cross multiple lines of **show interfaces** command output and will require the **-lineanchor** Tcl regex option to the Tcl **regexp** command. The `(\d+)/(\d+)/` character sequence matches and stores a variable that is composed of one or more decimal digits, followed by a forward slash (/) character, another variable composed of decimal digits, and then another / character. These variables represent the current and maximum hold input queue size.

**Script Implementation** 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 hold queue information. The variable must be set to `yes` (case ignored) for the syslog messages to be sent. A sample message that occurs when the `EEM_INTERFACE_INPUT_Q_DEBUG` variable is set to `yes` follows:

```
%HA_EM-7-LOG: interface-input-q.tcl: Interface GigabitEthernet0/0 input queu
```

Once the Tcl script has been tested, the `EEM_INTERFACE_INPUT_Q_DEBUG` variabl

The Tcl script can be copied to the vulnerable device, and if the Cisco IOS file system supports directories, a directory for EEM policies will be created. The following example copies the script to the **EEM** directory of the **DISK0:** file system:

```
router#mkdir DISK0:/EEM
Create directory filename [EEM]?
Created dir disk0:/EEM
router#copy <location of Tcl script> disk0:/EEM/interface-input-q.tcl
```

The Tcl script can then be registered as an EEM policy with the following global configuration commands:

```
!-- Define variable to 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
```

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

```
%HA_EM-7-LOG: interface-input-q.tcl: Interface GigabitEthernet0/0 input
queue full. 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 se
        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_qu
  }
}

# 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 NetFlow Records

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

```

router#show ip cache flow
IP packet size distribution (90784136 total packets):
  1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  48
  .000 .698 .011 .001 .004 .005 .000 .004 .000 .000 .003 .000 .000 .000 .00
    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
    .000 .001 .256 .000 .010 .000 .000 .000 .000 .000 .000

```

IP Flow Switching Cache, 4456704 bytes  
 1885 active, 63651 inactive, 59960004 added  
 129803821 aged polls, 0 flow alloc failures  
 Active flows timeout in 30 minutes  
 Inactive flows timeout in 15 seconds

IP Sub Flow Cache, 402056 bytes  
 0 active, 16384 inactive, 0 added, 0 added to flow  
 0 alloc failures, 0 force free  
 1 chunk, 1 chunk added  
 last clearing of statistics never

Protocol	Total Flows	Flows /Sec	Packets /Flow	Bytes /Pkt	Packets /Sec	Active(Sec) /Flow	Idle(Se) /Flow
TCP-Telnet	11393421	2.8	1	48	3.1	0.0	1.4
TCP-FTP	236	0.0	12	66	0.0	1.8	4.8
TCP-FTPD	21	0.0	13726	1294	0.0	18.4	4.1
TCP-WWW	22282	0.0	21	1020	0.1	4.1	7.3
TCP-X	840	0.0	1	40	0.0	0.0	2.9
TCP-BGP	1	0.0	1	40	0.0	0.0	15.0
TCP-Frag	70399	0.0	1	688	0.0	0.0	22.7
TCP-other	47861004	11.8	1	211	18.9	0.0	1.3
UDP-DNS	710	0.0	4	86	0.0	7.4	16.4
UDP-NTP	287252	0.0	1	76	0.0	0.0	15.5
UDP-other	310347	0.0	2	230	0.1	0.6	15.9
ICMP	11674	0.0	3	61	0.0	19.8	15.5
IPv6INIP	15	0.0	1	1132	0.0	0.0	15.4
GRE	4	0.0	1	48	0.0	0.0	15.3
Total:	59957957	14.8	1	196	22.5	0.0	1.5

SrcIf	SrcIPAddress	DstIf	DstIPAddress	Pr	SrcP	DstP	Pk
Gi0/0	192.168.10.160	Gi0/1	192.168.60.100	06	0984	13C4	
Gi0/0	192.168.11.54	Gi0/1	192.168.60.158	06	0911	13C5	
Gi0/1	192.168.150.60	Gi0/0	10.89.16.226	06	0016	12CA	
Gi0/0	192.168.13.97	Gi0/1	192.168.60.28	11	0B3E	13C4	
Gi0/0	192.168.12.185	Gi0/1	192.168.60.239	11	0BD7	07AF	
Gi0/0	192.168.21.115	Gi0/1	192.168.60.111	11	0BD7	09D5	
Gi0/0	192.168.88.235	Gi0/1	192.168.60.109	11	0BD7	097B	
Gi0/0	10.89.16.226	Gi0/1	192.168.150.60	06	12CA	0016	

router#

In the preceding example, there are multiple flows for **SIP packets on TCP port 5060 (hex value 13C4)**, **SIP packets on TCP port 5061 (hex value 13C5)**, **SIP packets on UDP port 5060 (hex value 13C4)**, **IP SLA Responder packets on UDP port 1967 (hex value 07AF)**, **H.323 Call Signaling Transport on UDP port 2517 (hex value 09D5)**, and **MGCP packets on UDP port 2427 (hex value 097B)**.

This traffic is sent to addresses within the 192.168.60.0/24 address block, which is used for infrastructure devices. The packets in these flows may be spoofed and may indicate an attempt to exploit these vulnerabilities. Administrators are advised to compare these flows to baseline utilization for the respective traffic sent on the following protocol and ports and also investigate the flows to determine whether they are sourced from untrusted hosts or networks:

- SIP using TCP port 5060
- SIP using TCP port 5061
- SIP using UDP port 5060
- IP SLA Responder using UDP port 1967
- H.323 Call Signaling Transport using UDP port 2517
- MGCP using UDP port 2427

To view only the traffic flows for the respective packets listed above, the command **show ip cache flow | include SrcIf|\_11\_.\*(13C4|07AF|09D5|097B)\_** and **show ip cache flow | include SrcIf|\_06\_.\*(13C4|13C5)\_** will display the related UDP and TCP NetFlow records as shown here:

### UDP Flows

```
router#show ip cache flow | include SrcIf|_11_.*(13C4|07AF|09D5|097B)_
```

SrcIf	SrcIPAddress	DstIf	DstIPAddress	Pr	SrcP	DstP	Pk
Gi0/0	192.168.13.97	Gi0/1	192.168.60.28	11	0B3E	13C4	
Gi0/0	192.168.12.185	Gi0/1	192.168.60.239	11	0BD7	07AF	
Gi0/0	192.168.21.115	Gi0/1	192.168.60.111	11	0BD7	09D5	
Gi0/0	192.168.88.235	Gi0/1	192.168.60.109	11	0BD7	097B	

router#

### TCP Flows

```
router#show ip cache flow | include SrcIf|_06_.*(13C4|13C5)_
```

SrcIf	SrcIPAddress	DstIf	DstIPAddress	Pr	SrcP	DstP	Pk
Gi0/0	192.168.10.160	Gi0/1	192.168.60.100	06	0984	13C4	
Gi0/0	192.168.11.54	Gi0/1	192.168.60.158	06	0911	13C5	

router#

## Cisco ASA, PIX, and FWSM Firewalls

### Mitigation: 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 these vulnerabilities when the attack originates from a trusted source address.

The tACL policy denies unauthorized packets that are sent to affected devices via the following protocols:

- SIP using TCP port 5060
- SIP using TCP port 5061
- SIP using UDP port 5060
- IP SLA Responder using UDP port 1967
- H.323 Call Signaling Transport using UDP port 2517
- MGCP using UDP port 2427

In the following example, 192.168.60.0/24 is the IP address space that is used by the affected devices, and the host at 192.168.100.1 is considered a trusted source that requires access to the affected devices. Care should be taken to allow required traffic for routing and administrative access prior to denying all unauthorized traffic.

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

```
!  
!-- Include any explicit permit statements for trusted sources  
!-- that require access on the vulnerable ports  
!  
access-list tACL-Policy extended permit tcp host 192.168.100.1 192.168.60.0  
access-list tACL-Policy extended permit tcp host 192.168.100.1 192.168.60.0  
access-list tACL-Policy extended permit udp host 192.168.100.1 192.168.60.0  
access-list tACL-Policy extended permit udp host 192.168.100.1 192.168.60.0  
access-list tACL-Policy extended permit udp host 192.168.100.1 192.168.60.0  
access-list tACL-Policy extended permit udp host 192.168.100.1 192.168.60.0  
  
!  
!-- The following vulnerability-specific access control entries  
!-- (ACEs) can aid in identification of attacks  
!  
access-list tACL-Policy extended deny tcp any 192.168.60.0 255.255.255.0 eq  
access-list tACL-Policy extended deny tcp any 192.168.60.0 255.255.255.0 eq  
access-list tACL-Policy extended deny udp any 192.168.60.0 255.255.255.0 eq  
access-list tACL-Policy extended deny udp any 192.168.60.0 255.255.255.0 eq  
access-list tACL-Policy extended deny udp any 192.168.60.0 255.255.255.0 eq  
access-list tACL-Policy extended deny udp any 192.168.60.0 255.255.255.0 eq  
  
!  
!-- 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  
!  
access-list tACL-Policy extended deny ip any any  
  
!  
!-- Apply tACL to interface(s) in the ingress direction
```

!

```
access-group tACL-Policy in interface outside
```

## Mitigation: Spoofing Protection Using Unicast Reverse Path Forwarding

The vulnerabilities that are described in this document 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 complete 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: Transit Access Control Lists

After the tACL has been applied to an interface, administrators can use the **show access-list** command to identify the number of higher layer protocol packets on their respective ports that have been filtered. Administrators are advised to investigate filtered packets to determine whether they are attempts to exploit these vulnerabilities. Example output for **show access-list tACL-Policy** follows:

```
firewall#show access-list tACL-Policy
access-list tACL-Policy; 13 elements
access-list tACL-Policy line 1 extended permit tcp host 192.168.100.1 192.16
access-list tACL-Policy line 2 extended permit tcp host 192.168.100.1 192.16
access-list tACL-Policy line 3 extended permit udp host 192.168.100.1 192.16
access-list tACL-Policy line 4 extended permit udp host 192.168.100.1 192.16
access-list tACL-Policy line 5 extended permit udp host 192.168.100.1 192.16
access-list tACL-Policy line 6 extended permit udp host 192.168.100.1 192.16
access-list tACL-Policy line 7 extended deny tcp any 192.168.60.0 255.255.25
access-list tACL-Policy line 8 extended deny tcp any 192.168.60.0 255.255.25
access-list tACL-Policy line 9 extended deny udp any 192.168.60.0 255.255.25
access-list tACL-Policy line 10 extended deny udp any 192.168.60.0 255.255.2
access-list tACL-Policy line 11 extended deny udp any 192.168.60.0 255.255.2
access-list tACL-Policy line 12 extended deny udp any 192.168.60.0 255.255.2
access-list tACL-Policy line 13 extended deny ip any any (hitcnt=156)
firewall#
```

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

- **21 SIP** packets on **TCP port 5060** for ACE line 7
- **29 SIP** packets on **TCP port 5061** for ACE line 8
- **22 SIP** packets on **UDP port 5060** for ACE line 9
- **9 IP SLA Responder** packets on **UDP port 1967** for ACE line 10
- **17 H.323 Call Signaling Transport** packets on **UDP port 2517** for ACE line 11
- **45 MGCP** packets on **UDP port 2427** for ACE line 12

## 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 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 in [Monitoring the Security Appliance - Configuring and Managing Logs](#). Information about configuring syslog on the FWSM for Cisco Catalyst 6500 Series switches and Cisco 7600 Series routers is in [Monitoring the Firewall Services Module](#).

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 potential attempts to exploit the vulnerabilities that are 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 in [Creating a Regular Expression](#).

```
firewall#show logging | grep 106023
Jan 30 2009 00:15:13: %ASA-4-106023: Deny tcp src outside:192.0.2.18/2944
dst inside:192.168.60.191/5060 by access-group "tACL-Policy"
Jan 30 2009 00:15:13: %ASA-4-106023: Deny tcp src outside:192.0.2.200/2945
dst inside:192.168.60.33/5061 by access-group "tACL-Policy"
Jan 30 2009 00:15:13: %ASA-4-106023: Deny udp src outside:192.0.2.99/2946
dst inside:192.168.60.240/5060 by access-group "tACL-Policy"
Jan 30 2009 00:15:13: %ASA-4-106023: Deny udp src outside:192.0.2.88/2949
dst inside:192.168.60.38/1967 by access-group "tACL-Policy"
Jan 30 2009 00:15:13: %ASA-4-106023: Deny udp src outside:192.0.2.92/2956
dst inside:192.168.60.47/2517 by access-group "tACL-Policy"
Jan 30 2009 00:15:13: %ASA-4-106023: Deny udp src outside:192.0.2.215/2967
dst inside:192.168.60.207/2427 by access-group "tACL-Policy"
firewall#
```

In the preceding example, the messages logged for the tACL *tACL-Policy* show potentially spoofed higher layer protocol packets for the following protocols and ports sent to the address block assigned to the affected devices:

- SIP using TCP port 5060
- SIP using TCP port 5061
- SIP using UDP port 5060
- IP SLA Responder using UDP port 1967
- H.323 Call Signaling Transport using UDP port 2517
- MGCP using UDP port 2427

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

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

### **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 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 in [Monitoring the Security Appliance - Configuring and Managing Logs](#). Information about configuring syslog on the FWSM for Cisco Catalyst 6500 Series switches and Cisco 7600 Series routers is in [Monitoring the Firewall Services Module](#).

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 potential attempts to exploit the vulnerabilities that are 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 in [Creating a Regular Expression](#).

```
firewall#show logging | grep 106021
Jan 30 2009 00:15:13: %ASA-1-106021: Deny UDP reverse path check from
192.168.60.1 to 192.168.60.100 on interface outside
Jan 30 2009 00:15:13: %ASA-1-106021: Deny UDP reverse path check from
192.168.60.1 to 192.168.60.100 on interface outside
Jan 30 2009 00:15:13: %ASA-1-106021: Deny TCP reverse path check from
192.168.60.1 to 192.168.60.100 on interface outside
```

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

```
firewall#show asp drop frame rpf-violated
Reverse-path verify failed          11
firewall#
```

In the preceding example, Unicast RPF has dropped **11 IP packets** received on interfaces with Unicast RPF configured. Absence of output indicates that the Unicast RPF feature on the firewall has not dropped packets.

For additional information about debugging accelerated security path dropped packets or connections, 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 1.0	2009-March-25	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](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](#)
- [Cisco Guide to Harden Cisco IOS Devices](#)
- [Cisco Security Center](#)
- [Cisco IOS NetFlow - Home Page on Cisco.com](#)
- [Cisco IOS NetFlow White Papers](#)
- [NetFlow Performance Analysis](#)
- [Cisco Network Foundation Protection White Papers](#)
- [Cisco Network Foundation Protection Presentations](#)
- [A Security-Oriented Approach to IP Addressing](#)
- [Understanding Control Plane Protection](#)
- [Cisco Firewall Products - Home Page on Cisco.com](#)
- [Unicast Reverse Path Forwarding Enhancements for the Internet Service Provider](#)
- [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](#)

© 2008 - 2009 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)