

# Characterizing and Tracing Packet Floods Using Cisco Routers

Document ID: 13609

---

## Introduction

### Prerequisites

- Requirements
- Components Used
- Conventions

### The Most Common DoS Attacks

#### A DoS Characterization Access List

- Smurf Ultimate Target
- Smurf Reflector
- Fraggle
- SYN Floods
- Other Attacks
- Logging and Counter Caveats

### Tracing

- Tracing With "log-input"
- SYN Flood
- Smurf Stimulus
- Tracing Without "log-input"

### Related Information

---

## Introduction

Denial of service (DoS) attacks are common on the Internet. The first step you use to respond to such an attack is to find out exactly what sort of attack it is. Many of the commonly used DoS attacks are based on high-bandwidth packet floods, or on other repetitive streams of packets.

The packets in many DoS attack streams can be isolated when you match them against Cisco IOS® software access list entries. This is valuable for filtering out attacks. It is also useful for when you characterize unknown attacks, and for when you trace "spoofed" packet streams back to their real sources.

Cisco router features such as debug logging and IP accounting can sometimes be used for similar purposes, especially with new or unusual attacks. However, with recent versions of Cisco IOS software, access lists and access list logging are the premiere features for when you characterize and trace common attacks.

## Prerequisites

### Requirements

There are no specific requirements for this document.

### Components Used

This document is not restricted to specific software and hardware versions.

## Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

## The Most Common DoS Attacks

A wide variety of DoS attacks are possible. Even if you ignore attacks that use software bugs to shut down systems with relatively little traffic, the fact remains that any IP packet that can be sent across the network can be used to execute a flooding DoS attack. When you are under attack, you must always consider the possibility that what you see is something that does not fall into the usual categories.

Subject to that caveat, however, it is also good to remember that many attacks are similar. Attackers choose common exploits because they are particularly effective, particularly hard to trace, or because tools are available. Many DoS attackers lack the skill or motivation to create their own tools, and use programs found on the Internet. These tools tend to fall in and out of fashion.

At the time of this writing, in July 1999, most customer requests for Cisco assistance involve the "smurf" attack. This attack has two victims: an "ultimate target" and a "reflector." The attacker sends a stimulus stream of ICMP echo requests ("pings") to the broadcast address of the reflector subnet. The source addresses of these packets are falsified to be the address of the ultimate target. For each packet sent by the attacker, many hosts on the reflector subnet respond. This floods the ultimate target and wastes bandwidth for both victims.

A similar attack, called "fraggle," uses directed broadcasts in the same way, but uses UDP echo requests instead of Internet Control Message Protocol (ICMP) echo requests. Fraggle usually achieves a smaller amplification factor than smurf, and is much less popular.

Smurf attacks are usually noticed because a network link becomes overloaded. A complete description of these attacks, and of defense measures, is on the Denial of Service Attacks Information page .

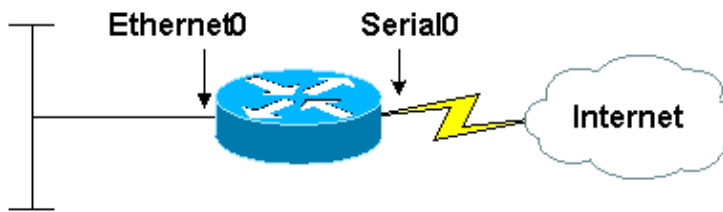
Another common attack is the SYN flood, in which a target machine is flooded with TCP connection requests. The source addresses and source TCP ports of the connection request packets are randomized. The purpose is to force the target host to maintain state information for many connections that are never completed.

SYN flood attacks are usually noticed because the target host (frequently an HTTP or SMTP server) becomes extremely slow, crashes, or hangs. It is also possible for the traffic that returns from the target host to cause trouble on routers. This is because this return traffic goes to the randomized source addresses of the original packets, it lacks the locality properties of "real" IP traffic, and can overflow route caches. On Cisco routers, this problem often manifests itself in the router running out of memory.

Together, smurf and SYN flood attacks account for the vast majority of the flooding DoS attacks reported to Cisco, and recognizing them quickly is very important. Both attacks (as well as some "second tier" attacks, such as ping floods) are easily recognized when you use Cisco access lists.

## A DoS Characterization Access List

Picture a router with two interfaces. Ethernet 0 is connected to an internal LAN at a business or small ISP. Serial 0 provides an Internet connection via an upstream ISP. The input packet rate on serial 0 is "pegged" at the full link bandwidth, and hosts on the LAN run slowly, crash, hang, or show other signs of a DoS attack. The small site at which the router connects has no network analyzer, and the people there have little or no experience in reading analyzer traces even if the traces are available.



### 10.2.3.x network

Now, assume that you apply an access list as this output shows:

```
access-list 169 permit icmp any any echo
access-list 169 permit icmp any any echo-reply
access-list 169 permit udp any any eq echo
access-list 169 permit udp any eq echo any
access-list 169 permit tcp any any established
access-list 169 permit tcp any any
access-list 169 permit ip any any

interface serial 0
ip access-group 169 in
```

This list does not filter out any traffic at all; all the entries are permits. However, because it categorizes packets in useful ways, the list can be used to tentatively diagnose all three types of attacks: smurf, SYN floods, and fraggle.

## Smurf Ultimate Target

If you issue the **show access-list** command, you see output similar to this:

```
Extended IP access list 169
  permit icmp any any echo (2 matches)
  permit icmp any any echo-reply (21374 matches)
  permit udp any any eq echo
  permit udp any eq echo any
  permit tcp any any established (150 matches)
  permit tcp any any (15 matches)
  permit ip any any (45 matches)
```

Most of the traffic that arrives on the serial interface consists of ICMP echo reply packets. This is probably the signature of a smurf attack, and our site is the ultimate target, rather than the reflector. You can gather more information about the attack when you revise the access list, as this output shows:

```
interface serial 0
no ip access-group 169 in

no access-list 169
access-list 169 permit icmp any any echo
access-list 169 permit icmp any any echo-reply log-input
access-list 169 permit udp any any eq echo
access-list 169 permit udp any eq echo any
access-list 169 permit tcp any any established
access-list 169 permit tcp any any
access-list 169 permit ip any any

interface serial 0
ip access-group 169 in
```

The change here is that the **log-input** keyword is added to the access list entry that matches the suspect traffic. (Cisco IOS Software Releases earlier than 11.2 lack this keyword. Use the "log" keyword instead.) This causes the router to log information about packets that match the list entry. If you assume that **logging buffered** is configured, you can see the messages that result with the **show log** command (it can take a while for the messages to accumulate because of rate limiting). The messages appear similar to this output:

```
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.142
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.113
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.72
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.154
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.15
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.142
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.47
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.35
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.113
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.59
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.82
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.56
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.84
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.47
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.35
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet

%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.15
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.33
(Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1 packet
```

The source addresses of the echo reply packets are clustered in address prefixes 192.168.212.0/24, 192.168.45.0/24, and 172.16.132.0/24. (Private addresses in the 192.168.x.x and 172.16.x.x networks would not be on the Internet; this is a lab illustration.) This is very characteristic of a smurf attack, and the source addresses are the addresses of the smurf reflectors. If you look up the owners of these address blocks in the appropriate Internet "whois" databases, you can find the administrators of these networks, and ask for their help in dealing with the attack.

It is important at this point in a smurf incident to remember that these reflectors are fellow victims, not attackers. It is extremely rare for attackers to use their own source addresses on IP packets in any DoS flood, and impossible for them to do so in a working smurf attack. Any address in a flood packet should be assumed to be either completely falsified, or the address of a victim of some sort. The most productive approach for the

ultimate target of a smurf attack is to contact the reflectors, either to ask them to reconfigure their networks to shut down the attack, or to ask for their assistance in tracing the stimulus stream.

Because the damage to the ultimate target of a smurf attack is usually caused by overloading of the incoming link from the Internet, there is often no response other than to contact the reflectors. By the time the packets arrive at any machine under the control of the target, most of the damage has already been done.

One stopgap measure is to ask the upstream network provider to filter out all ICMP echo replies, or all ICMP echo replies from specific reflectors. It is not recommended that you leave this sort of filter in place permanently. Even for a temporary filter, only echo replies should be filtered, not all ICMP packets. Another possibility is to have the upstream provider use Quality of Service and rate limiting features to restrict the bandwidth available to echo replies. A reasonable bandwidth limitation can be left in place indefinitely. Both of these approaches depend on the equipment of the upstream provider having the necessary capacity, and sometimes that capacity is not available.

## Smurf Reflector

If the incoming traffic consists of echo requests rather than echo replies (in other words, if the first access list entry, rather than the second, was counting many more matches than could reasonably be expected), you would suspect a smurf attack in which the network was being used as a reflector, or possibly a simple ping flood. In either case, if the attack is a success, you would expect the outgoing side of the serial line to be swamped, as well as the incoming side. In fact, because of the amplification factor, you would expect the outgoing side to be even more overloaded than the incoming side.

There are several ways to distinguish the smurf attack from the simple ping flood:

- Smurf stimulus packets are sent to a directed broadcast address, rather than to a unicast address, whereas ordinary ping floods almost always use unicasts. You can see the addresses that use the **log-input** keyword on the appropriate access list entry.
- If you are used as a smurf reflector, there is a disproportionate number of output broadcasts in the **show interface** display on the Ethernet side of the system, and usually a disproportionate number of broadcasts sent in the **show ip traffic** display. A standard ping flood does not increase the background broadcast traffic.
- If you are used as a smurf reflector, there is more traffic outgoing towards the Internet than traffic incoming from the Internet. In general, there is more output packets than input packets on the serial interface. Even if the stimulus stream completely fills the input interface, the response stream is larger than the stimulus stream, and packet drops are counted.

A smurf reflector has more options than the ultimate target of a smurf attack. If a reflector chooses to shut down the attack, appropriate use of **no ip directed-broadcast** (or equivalent non-IOS commands) usually suffices. These commands belong in every configuration, even if there is no active attack. For more information on the prevention of your Cisco equipment from being used in a smurf attack, refer to *Improving Security on Cisco Routers*. For more general information about smurf attacks in general, and for information about protecting non-Cisco equipment, refer to the *Denial of Service Attacks Information* page .

A smurf reflector is one step closer to the attacker than is the ultimate target, and is therefore in a better position to trace the attack. If you choose to trace the attack, you need to work with the ISPs involved. If you wish to have any action taken when you complete the trace, you need to work with appropriate law enforcement agencies. If you seek to trace an attack, it is recommended that you involve law enforcement as soon as possible. See the *Tracing* section for technical information on tracing flooding attacks.

## Fraggle

The fraggle attack is analogous to the smurf attack, except that UDP echo requests are used for the stimulus stream instead of ICMP echo requests. The third and fourth lines of the access list identify fraggle attacks. The appropriate response for the victims is the same, except that UDP echo is a less important service in most networks than is ICMP echo. Therefore, you can disable them completely with fewer negative consequences.

## SYN Floods

The fifth and sixth lines of the access list are:

```
access-list 169 permit tcp any any established
access-list 169 permit tcp any any
```

The first of these lines matches any TCP packet with the ACK bit set. For our purposes, what this really means is that it matches any packet that is not a TCP SYN. The second line matches only packets that are TCP SYNs. A SYN flood is easily identified from the counters on these list entries. In normal traffic, non-SYN TCP packets outnumber SYNs by at least a factor of two, and usually more like four or five. In a SYN flood, SYNs typically outnumber non-SYN TCP packets many times over.

The only non-attack condition that creates this signature is a massive overload of genuine connection requests. In general, such an overload will not come unexpectedly, and will not involve as many SYN packets as a real SYN flood. Also, SYN floods often contain packets with completely invalid source addresses; using the **log-input** keyword, it is possible to see if connection requests come from such addresses.

There is an attack called a "process table attack" which bears some similarity to the SYN flood. In the process table attack, the TCP connections are completed, then allowed to time out with no further protocol traffic, whereas in the SYN flood, only the initial connection requests are sent. Because a process table attack requires the completion of the TCP initial handshake, it must generally be launched with the use of the IP address of a real machine to which the attacker has access (usually stolen access). Process table attacks are therefore easily distinguished from SYN floods with the use of packet logging. All the SYNs in a process table attack come from one or a few addresses, or at the most from one or a few subnets.

Response options for the victims of SYN floods are very limited. The system under attack is usually an important service, and blocking access to the system usually accomplishes what the attacker wants. Many router and firewall products, including Cisco's, have features that can be used to reduce the impact of SYN floods. But, the effectiveness of these features depends on the environment. For more information, refer to the documentation for the Cisco IOS Firewall Feature Set, the documentation for the Cisco IOS TCP Intercept feature, and Improving Security on Cisco Routers.

It is possible to trace SYN floods, but the tracing process requires the assistance of each ISP along the path from the attacker to the victim. If you decide to try to trace a SYN flood, contact law enforcement early on, and work with your own upstream service provider. See the tracing section of this document for details on tracing with the use of Cisco equipment.

## Other Attacks

If you believe that you are under an attack, and if you can characterize that attack using IP source and destination addresses, protocol numbers, and port numbers, you can use access lists to test your hypothesis. Create an access list entry that matches the suspect traffic, apply it to an appropriate interface, and either watch the match counters or log the traffic.

## Logging and Counter Caveats

The counter on an access list entry counts all matches against that entry. If you apply an access list to two interfaces, the counts you see are aggregate counts.

Access list logging does not show every packet that matches an entry. Logging is rate-limited to avoid CPU overload. What logging shows you is a reasonably representative sample, but not a complete packet trace. Remember that there are packets that you do not see.

In some software versions, access list logging works only in certain switching modes. If an access list entry counts a lot of matches, but logs nothing, try to clear the route cache to force packets to be process switched. Be careful if you do this on heavily loaded routers with many interfaces. A lot of traffic can get dropped while the cache is rebuilt. Use Cisco Express Forwarding whenever possible.

Access lists and logging have a performance impact, but not a large one. Be careful on routers that run at more than about 80 percent CPU load, or when you apply access lists to very high-speed interfaces.

## Tracing

The source addresses of DoS packets are almost always set to values that have nothing to do with the attackers themselves. Therefore, they are not useful in the identification of the attackers. The only reliable way to identify the source of an attack is to trace it back hop-by-hop through the network. This process involves the reconfiguration of routers and the examination of log information. Cooperation by all network operators along the path from the attacker to the victim is required. Securing that cooperation usually requires the involvement of law enforcement agencies, who must also be involved if any action is to be taken against the attacker.

The tracing process for DoS floods is relatively simple. Starting at a router (named "A") that is known to be carrying flood traffic, one identifies the router (named "B") from which A is receiving the traffic. One then logs into B, and finds the router (named "C") from which B is receiving the traffic. This continues until the ultimate source is found.

There are several complications in this method, which this list describes:

- The "ultimate source" can be a computer which has been compromised by the attacker, but which is actually owned and operated by another victim. In this case, tracing the DoS flood is only the first step.
- Attackers know that they can be traced, and usually continue their attacks only for a limited time. There may not be enough time to actually trace the flood.
- Attacks can come from multiple sources, especially if the attacker is relatively sophisticated. It is important to try to identify as many sources as possible.
- Communication problems slow down the tracing process. Frequently one or more of the network operators involved does not have appropriately skilled staff available.
- Legal and political concerns may make it difficult to act against attackers even if one is found.

Most efforts to trace DoS attacks fail. Because of this, many network operators do not even attempt to trace an attack unless placed under pressure. Many others trace only "severe" attacks, with differing definitions of what is "severe." Some assist with a trace only if law enforcement is involved.

## Tracing With "log-input"

If you choose to trace an attack that passes through a Cisco router, the most effective way to do this is to construct an access list entry that matches the attack traffic, attach the **log-input** keyword to it, and apply the access list outbound on the interface through which the attack stream is sent toward its ultimate target. The log

entries produced by the access list identify the router interface through which the traffic arrives, and, if the interface is a multipoint connection, give the Layer 2 address of the device from which it is received. The Layer 2 address can then be used to identify the next router in the chain, using, for example, the **show ip arp mac-address** command.

## SYN Flood

In order to trace a SYN flood, you can create an access list similar to this:

```
access-list 169 permit tcp any any established
access-list 169 permit tcp any host victim-host log-input
access-list 169 permit ip any any
```

This logs all SYN packets destined for the target host, including legitimate SYNs. In order to identify the most likely actual path toward the attacker, examine the log entries in detail. In general, the source of the flood is the source from which the largest number of matching packets arrives. The source IP addresses themselves mean nothing. You are looking for source interfaces and source MAC addresses. Sometimes it is possible to distinguish flood packets from legitimate packets because flood packets can have invalid source addresses. Any packet whose source address is not valid is likely to be part of the flood.

The flood can come from multiple sources, although this is relatively unusual for SYN floods.

## Smurf Stimulus

In order to trace a smurf stimulus stream, use an access list like this:

```
access-list 169 permit icmp any any echo log-input
access-list 169 permit ip any any
```

Note that the first entry does not restrict itself to packets destined for the reflector address. The reason for this is that most smurf attacks use multiple reflector networks. If you are not in contact with the ultimate target, you may not know all the reflector addresses. As your trace gets closer to the source of the attack, you may begin to see echo requests going to more and more destinations; this is a good sign.

However, if you deal with a great deal of ICMP traffic, this can generate too much logging information for you to read easily. If this happens, you can restrict the destination address to be one of the reflectors that is known to be used. Another useful tactic is to use an entry that takes advantage of the fact that netmasks of 255.255.255.0 are very common in the Internet. And, because of the way that attackers find smurf reflectors, the reflector addresses actually used for smurf attacks are even more likely to match that mask. Host addresses that end in .0 or .255 are very uncommon in the Internet. Therefore, you can build a relatively specific recognizer for smurf stimulus streams as this output shows:

```
access-list 169 permit icmp any host known-reflector echo log-input
access-list 169 permit icmp any 0.0.0.255 255.255.255.0 echo log-input
access-list 169 permit icmp any 0.0.0.0 255.255.255.0 echo log-input
access-list 169 permit ip any any
```

With this list, you can eliminate many of the "noise" packets from your log, while you still have a good chance of noticing additional stimulus streams as you get closer to the attacker.

## Tracing Without "log-input"

The **log-input** keyword exists in Cisco IOS Software Releases 11.2 and later, and in certain 11.1-based software created specifically for the service provider market. Older software does not support this keyword. If you use a router with older software, you have three viable options:

- Create an access list without logging, but with entries that match the suspect traffic. Apply the list on the *input* side of each interface in turn, and watch the counters. Look for interfaces with high match rates. This method has a very small performance overhead, and is good for the identification of source interfaces. Its biggest drawback is that it does not give link-layer source addresses, and is therefore useful mostly for point-to-point lines.
- Create access list entries with the **log** keyword (as opposed to **log-input**). Once again, apply the list to the incoming side of each interface in turn. This method still does not give source MAC addresses, but can be useful for seeing IP data. For instance, to verify that a packet stream really is part of an attack. Performance impact can be moderate to high and newer software performs better than older software.
- Use the **debug ip packet detail** command to collect information about packets. This method gives MAC addresses, but can have serious performance impact. It is easy to make a mistake with this method and make a router unusable. If you use this method, make sure that the router switches the attack traffic in fast, autonomous, or optimum mode. Use an access list to restrict debugging to only the information you really need. Log debugging information to the local log buffer, but turn off logging of debug information to Telnet sessions and to the console. If possible, arrange for someone to be physically near the router, so that it can be power cycled as necessary.

Remember that the **debug ip packet** command does not display information about fast-switched packets. You need to issue the **clear ip cache** command in order to capture information. Each **clear** command gives you one or two packets of debug output.

---

## Related Information

- [Kerberos](#)
- [Technical Support & Documentation – Cisco Systems](#)

---

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

© 2009 – 2010 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

---

Updated: May 27, 2005

Document ID: 13609

---