Security testing with hping

AT THE HOP

Don't let intruders crash your dance. We'll show you how to test your firewalls and intrusion detection systems with hping.

BY JAMES STANGER, PHD

When it comes to penetration testing and security audits, hping is one of your best friends. Currently in its third iteration, hping has become a preferred way to generate IP packets, usually for the purpose of testing firewall and intrusion detection systems.

Because you can use hping to manipulate all of the fields, flags, and protocol types of the TCP/IP protocol suite, some users call it a packet-crafting application.

By manipulating packets, you can scan systems stealthily, generate traffic floods, and generally create packets to your heart's content. Over the years, hping has become the de facto packet generator.

Generating custom packets is nothing new. Previous tools with whiz-bang and hackerish names, such as targa, synful, papa smurf, and netdude, could help with the task of generating designer packets, but many of these older applications had problems and limitations. For example, some tools could only scan Class C IPv4 networks.

What Does hping Do?

Hping provides a single, universal solution that helps prevent many problems of the previous generation. Hping is designed to:

- scan hosts,
- assist with penetration testing,
- test intrusion detection systems,

Versions

Hping3 is the latest version of hping, and hping2 is the most significant predecessor application. Several applications depend upon hping2, which has been around quite a bit longer than hping3.

I install both versions, and I recommend that you do the same. I use hping3 as a stand-alone application, but I still have hping2 in case I need it for third-party applications, such as scapy (another packet manipulation tool) and idswakeup (an application for auditing intrusion detection systems). Hping3 comes with a new TCL scripting engine and is, therefore, quite bit more powerful than a simple command-line tool.

The original hping and hping2 applications operate as one-time commands – they don't launch an interactive shell. If you use the command without any arguments, hping3 places you into a session, much like the old nslookup command.

Hping3 lets you create fairly sophisticated scripts that will help you simulate traffic for your firewalls and intrusion detection systems. A less obvious advantage of hping3 is that Salvatore Sanfilippo, the creator of all things hping, rewrote much of the underlying code.

- and send files between hosts.

In this article, I will explain how to start generating test packets with hping.

Installing hping

Hping3 is available from the project website as a source tarball [1]. If you're using an Ubuntu or Debian system, you can use either Synaptic Package Manager or apt-get for the installation. To install hping, enter the following command:

```
sudo apt-get ➥
install hping3.
```

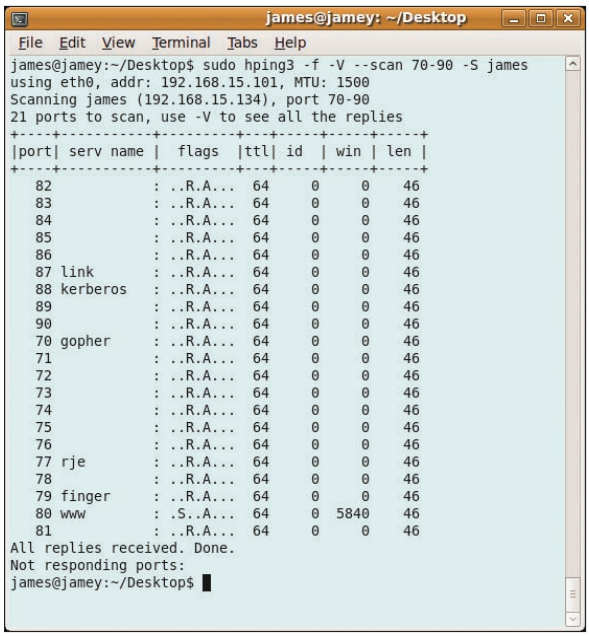

Figure 1: Viewing a packet generated by hping in Wireshark.

You don't need to enable any additional repositories. Red Hat or CentOS packages are also available online [2].

Scanning Hosts

After installing hping, you are ready to get started. Suppose you want to send two TCP packets to a system named *james*, and you want those packets to hit port 80 on *james*. To do this, you would issue the command shown (with the accompanying output) in Listing 1.

In Listing 1, notice that the *flags* = field is set to *SA*, which is hping's way of telling you that port 80 is open on *james*. If the ports were closed, you'd see *RA* in the *flags* = field.

The *-S* option sends a SYN packet, which often is used to create scans that are hard for intrusion detection systems to detect and flag as threatening.

After a system replies to a SYN packet, you know that a port is listening; the intrusion detection system will treat the SYN packet as standard traffic rather than as a threat.

Figure 1 shows how to specify a more sophisticated scan that provides a nice little ASCII-based report.

Advantages Over Nmap

You might wonder why you would want to use hping to look for open ports when you already have Nmap. In some situations, hping offers advantages over Nmap.

First, hping is a lightweight application; if you've got it installed and ready to go, why worry about installing anything more?

Second, it's always good to know how to do the same thing with more than one application.

Listing 1: A Simple Scan

```
01 pink@floyd:~/Desktop$ sudo hping3 -S james -c 2 -p 80
02 HPING james (eth0 192.168.15.134): S set, 40 headers + 0 data
   bytes
03 len=46 ip=192.168.15.134 ttl=64 DF id=0 sport=80 flags=SA seq=0
   win=5840 rtt=0.3 ms
04 len=46 ip=192.168.15.134 ttl=64 DF id=0 sport=80 flags=SA seq=1
   win=5840 rtt=0.3 ms
05
06 --- james hping statistic ---
07 2 packets transmitted, 2 packets received, 0% packet loss
08 round-trip min/avg/max = 0.3/0.3/0.3 ms
09 pink@floyd:~/Desktop$
```

Hping's creator, for example, still maintains the tool even though he's collaborated for years with Fyodor, the creator of Nmap.

Third, you can also conduct incremental scans, which means each scan will climb up one port on a system:

```
sudo hping -S targethost-p ++0
```

This command creates a report that tells you what ports are open on the system.

A Better Traceroute?

One interesting feature of hping3 is that you can generate a more revealing traceroute report using any protocol. For example, suppose you want to determine exactly what happens on each hop of a traceroute. To do this, you can specify the use of a TCP SYN packet. The *-T* option allows you to enable hping3's traceroute function. In the command shown above, the *--ttl* option allows you to specify the number of routers (i.e., hops) you want to transmit.

If you want to issue a traceroute command using UDP, the command shown in Listing 2 will suffice. The output shows how each router processes the UDP packet.

Why would you want to do such a thing? Because many routers block traditional ICMP packets, even if your latest system used UDP.

Listing 2: Tracing UDP

```
01 sudo hping3 -2 192.168.44.45 -p ++44444 -T -n
02
03 HPING 192.168.44.45 (eth0 192.168.44.45): udp mode
   set, 28 headers + 0 data bytes
04 hop=1 TTL 0 during transit from ip=172.16.8.1
05 hop=1 hoprtt=1.7 ms
06 hop=2 TTL 0 during transit from ip=12.155.83.1
07 hop=2 hoprtt=2.7 ms
08 hop=3 TTL 0 during transit from ip=12.119.43.49
09 hop=3 hoprtt=10.0 ms
10 hop=4 TTL 0 during transit from ip=12.123.21.30
11 hop=4 hoprtt=13.6 ms
12 hop=5 TTL 0 during transit from ip=12.122.12.21
13 hop=5 hoprtt=13.3 ms
14 hop=6 TTL 0 during transit from ip=12.122.17.42
15 hop=6 hoprtt=11.9 ms
16 hop=7 TTL 0 during transit from ip=12.122.96.9
17 hop=7 hoprtt=36.6 ms
18 hop=8 TTL 0 during transit from ip=192.205.34.62
19 hop=8 hoprtt=13.6 ms
20 hop=9 TTL 0 during transit from ip=4.68.103.46
```

To analyze one particular hop of a traceroute packet, you can use the *--tr-keep-ttl* option:

```
sudo hping3 -S 12.119.80.1 ➥
-p 80 -T --ttl 3 ➥
--tr-keep-ttl -n
```

The *-n* option ensures that numbers aren't resolved.

The preceding command issues TCP-based packets to the target host, but then reports only the third hop. The output is shown in Listing 3. The information in Listing 3 can help you determine exactly if and how a particular router is altering packets in transit.

Discovering the MTU

To determine the MTU (Maximum Transmission Unit – the largest datagram allowed for the network), you could issue the following command:

```
hping3 -D -V -I em1 ➥
--icmp targethost
```

Replace *targethost* with the host name or IP address of the system on the network where you want to test the MTU.

Why is it important to discover the MTU? First, VPN connections and other network transmissions sometimes encounter problems if the MTU on a system or a network is set strangely.

In convergence networks (for example, where you're implementing a VoIP SIP or H.323 system), you might need to determine the MTU to avoid problems with jitter and traffic congestion. By determining the MTU and adjusting it properly at the router or your individual hosts, you can reduce latency and resolve call quality issues that would otherwise prove elusive.

Listing 3: Analyzing a Hop

```
01 hop=3 TTL 0 during transit from ip=12.119.43.61
02 hop=3 hoprtt=31.7 ms
03 hop=3 TTL 0 during transit from ip=12.119.43.61
04 hop=3 hoprtt=6.9 ms
05 hop=3 TTL 0 during transit from ip=12.119.43.61
06 hop=3 hoprtt=5.0 ms
07 hop=3 TTL 0 during transit from ip=12.119.43.49
08 hop=3 hoprtt=5.2 ms
09 hop=3 TTL 0 during transit from ip=12.119.43.49
10 hop=3 hoprtt=5.2 ms
11 hop=3 TTL 0 during transit from ip=12.119.43.49
12 hop=3 hoprtt=4.9 ms
13 hop=3 TTL 0 during transit from ip=12.119.43.61
14 hop=3 hoprtt=5.4 ms
15 hop=3 TTL 0 during transit from ip=12.119.43.61
```

Perimeter Testing

Perimeter testing means determining exactly what your firewall blocks and what it allows. To conduct a good test, you can spoof source IP addresses and source ports:

```
sudo hping3 -a ➥
10.0.44.45 -S james -c 2 -p 80
```

The result of the above command is that packets will appear to originate from the system at 10.0.44.45. Such a packet is useful for determining whether the firewall is allowing random packets in or out of your network.

In these cases, you don't have to use TCP. Using hping, you can generate UDP packets as well:

```
sudo hping3 targethost -c 2 ➥
--udp --baseport 80 ➥
--destport 80
```

The preceding command sends two UDP packets to port 80 on the target system from port 80 of your own system.

Of course, you can spoof the source IP address as well as the originating and destination ports:

```
sudo hping3 localhost -a ➥
10.0.44.45 -c 2 --udp ➥
--baseport 80 --destport 80
```

Sending Files

Creating a tunnel is one way to find out what your firewall is capable of blocking. On your receiving system, issue the following command:

```
host$ sudo hping3 -i eth0 ➥
--listen signature --icmp
```

To send the contents of the file on your local system to a remote system named *james*, issue the following command:

Penetration Testing

It's not enough to just know about how to use hping3; you need to also understand the basics of a penetration test. A typical test includes the following basic steps:

- **Network resource identification**: This step is sometimes called *network mapping*, *network footprinting*, or *target identification*. The step involves scanning systems for open ports, fingerprinting operating systems, and determining the types of applications that are operating on open ports.
- **Scanning for vulnerabilities**: Looking for vulnerabilities on server, firewall, and VoIP operating systems. You also conduct tests designed to break the existing authentication scheme. Once you are finished cracking systems, you then prioritize resources you have identified. For example, a system may have a fairly serious vulnerability that might not be very important. You might need to actually assign this system a lower priority than others that are considered more vital, especially if the vulnerable system isn't likely to become a stage for an attack. Many times, this step is considered part of the network resource identification, but I like to treat this activity as something separate. Determining vulnerabilities is a complex task that requires quite a bit of analytical thought.
- **Perimeter testing**: A classic activity for hping3. For example, you can use hping3 to generate traffic that tests whether the firewall is capable of blocking spoofed internal packets.
- **Intrusion detection testing**: In this step, you generate traffic to see if the intrusion detection system is capable of identifying anomalies and problems. Applications such as hping3 are perfect for generating such anomalous traffic.
- **Consideration of security policy and end user issues**: In this step, you determine the effectiveness of the security policy, and how well the network's applications ensure compliance. You also determine how well end users comply with the security policy. Although this last step isn't really relevant to applications such as hping3, it's important to understand that an auditor does more than scan systems and generate packets.

```
user@host:~$ sudo hping3 -I ➥
eth0 localhost --icmp -d 100 ➥
--sign signature ➥
--file /etc/shadow
```

On your receiving system's terminal, you will see the output of the file you're sending (see Listing 4).

Notice that the contents of the file has been sent through the firewall. Also notice that I've decided to send the contents of a particularly sensitive file. Creating an ad-hoc tunnel in this way allows quick file transfer back and forth across a firewall. Furthermore, this feature is useful for testing exactly what a firewall is capable of blocking.

Simulating Attacks

The LAND attack [4], which first appeared in 1997, involves sending a spoofed packet with its SYN flag activated to a target host. This spoofed packet has the same source IP and source port as the target hosts's IP. When the attack first appeared, it caused unpatched Windows systems (and some Linux systems) to create an infinite connection loop and crash.

Many attackers exploited this bug to wage simple, sophomoric, and highly annoying denial of service attacks. More sophisticated users realized that such attacks were useful for hijacking attacks.

A new variation of the LAND attack turned up in 2005, and this classic technique could easily appear again.

Hping3 can help you ensure that your systems are immune to such an attack. Suppose you want to test a system with the IP address of 192.168.2.3 that has port 139 open. To do so, you would issue the following command:

Choosing an Audit Type

At the risk of oversimplifying, two types of audits exist: blind and non-blind. A blind audit is one in which you adopt the perspective of a hacker who doesn't know about the network and has to discover all of the systems. With non-blind audit, you don't need to worry about discovering the systems; instead, you focus on scanning the systems for vulnerabilities. Regardless of the approach you take to auditing, your goal is to discover resources, show how to penetrate the defenses, and demonstrate how an attack could spread to other systems.

```
sudo hping3 -S 192.168.2.3 ➥
-a 192.168.2.3 -k -s 139 ➥
-p 139 --flood
```

This attack could cause an unpatched target system to freeze. Also notice the *--flood* option, which sends thousands of packets to the system.

Firewalls and Session State

Suppose you want to determine how well your firewall is able to record requests for Microsoft protocols across the network. To use hping3 to generate the packets for this test, issue the following commands:

```
hping www.acme.net -S -c 1 -p 139
hping www.acme.net -S -A -c 1 -p 139
hping www.acme.net -S -A -c 1 -p 135
```

These commands generate packets that the firewall – if its capability for maintaining state is working – will record. To verify this, you'll need to check the firewall's logs and use a packet sniffer.

Christmas Tree Packet

A Christmas tree packet [5] is a TCP packet that has almost every flag set, which is useful for bypassing firewalls and for launching various other forms of attack.

To create a Christmas tree packet using hping3, issue the following command:

```
hping3 -F -P -U ➥
10.44.45.15 -p 0
```

Firewalls and Time Stamps

In many cases, a firewall will automatically drop packets that don't have a time stamp. To add a time stamp to your packets, use the *-timestamp* option in your command:

```
hping3 -S 72.14.207.99 ➥
-p 80 --tcp-timestamp
```

The results will help you determine whether you need to enable timestamp filtering on the firewall.

Conclusion

Conducting penetration tests and determining the effectiveness of intrusion detection systems involves a great deal of skill and patience. Also, you need the right kind of tools. Hping3 is one of those tools. In this article, I've outlined only a few of the sophisticated commands at your disposal when you add hping to your security-testing toolkit. ■

Listing 4: Sending a File

```
01 Warning: Unable to guess the output
   interface
02 hping3 listen mode
03 [main] memlockall(): Success
04 Warning: can't disable memory
   paging!
05 99999:7:::
06 proxy:*:14181:0:99999:7:::
07 www-data:*:14181:0:99999:7:::
08 backup:*:14181:0:99999:99999:7:::
09 proxy:*:14181:0:99999:7:::
10 www-data:*:14181:0:99999:7:::
11 backup:*:14181:0:99999:7:::
12 list:*:14181:0:99999:7:::
13 irc:*:14181:0:99999:7:::
14 gnats:*:14181:0:99999:7:::
15 nobody:*7:::
16 nobody:*^C
17 [code snipped due to hitting Ctrl +
   C to end the transmission]
18 ---  hping statistic ---
19 0 packets transmitted, 0 packets
   received, 0% packet loss
20 round-trip min/avg/max =
   0.0/0.0/0.0 ms
```

INFO

[1] hping Project: *www.hping.org*

[2] hping in RPM: *http://dag.wieers.com/rpm/packages/hping*

[3] Packet crafting via hping: *http://www.governmentsecurity.org/archive/t835.html*

[4] LAND attack: *http://en.wikipedia.org/wiki/LAND*

[5] Christmas tree packet: *http://en.wikipedia.org/wiki/Christmas_tree_packett835.html*

THE AUTHOR

Dr. James Stanger is an accomplished writer, security consultant, and web designer. Currently, he is Vice President of Certification for the Certified Internet Web Professional assessment program and chair of the Linux Professional Institute (LPI) Advisory Council. Dr. Stanger is a co-author of O'Reilly's *LPI Certification in a Nutshell*. He runs a blog at http://*www.ciwcommunity.org*.