PING AND SOME OF THE THINGS IT CAN BE USED FOR
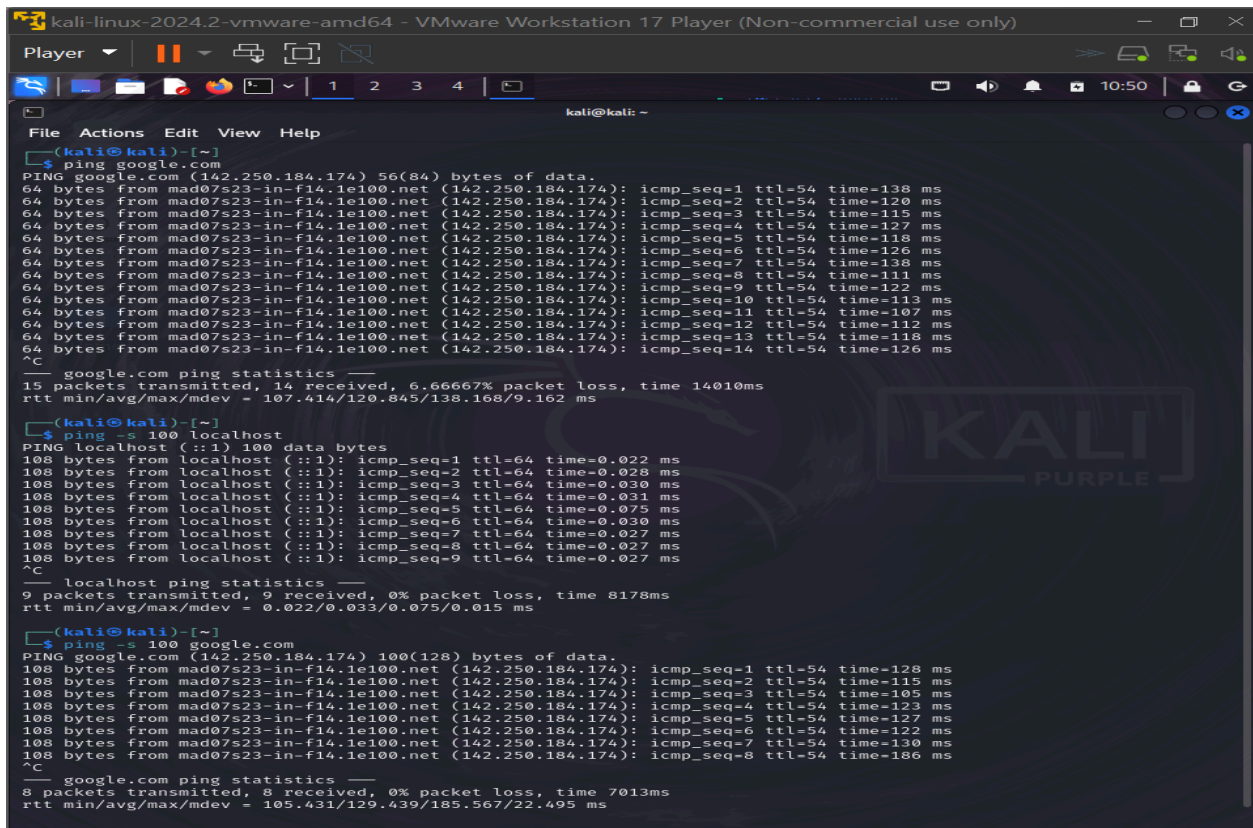
Tools : KALI LINUX [PING]
Project-Site : google.com, cisco.com

PING is a simple and useful network-based utility which can be used to identify if a host is alive or dead. Technically, we can call it an echo reply. By "alive", I mean that the host is active, and by "dead", that the host is in shutdown mode. Anything which has a network card can be a host: computers, servers, switches, websites, smartphones, IOT devices, etc.

It is often useful when setting up some new infrastructure to use ping to test if your infrastructure can correctly reach the network.

Input from Kali : PING

Player ▾

1 2 3 4

10:53

kali@kali: ~

File Actions Edit View Help

```
8 packets transmitted, 8 received, 0% packet loss, time 7013ms
rtt min/avg/max/mdev = 105.431/129.439/185.567/22.495 ms
  ┌──(kali㉿kali)-[~]
  └─$ ping -c 5 cisco.com
PING cisco.com (72.163.4.185) 56(84) bytes of data.
64 bytes from redirect-ns.cisco.com (72.163.4.185): icmp_seq=5 ttl=234 time=275 ms

── cisco.com ping statistics ──
5 packets transmitted, 1 received, 80% packet loss, time 4081ms
rtt min/avg/max/mdev = 274.532/274.532/274.532/0.000 ms

  ┌──(kali㉿kali)-[~]
  └─$ ping -c 5 cisco.com
PING cisco.com (72.163.4.185) 56(84) bytes of data.
64 bytes from redirect-ns.cisco.com (72.163.4.185): icmp_seq=1 ttl=234 time=242 ms
64 bytes from redirect-ns.cisco.com (72.163.4.185): icmp_seq=2 ttl=234 time=238 ms
64 bytes from redirect-ns.cisco.com (72.163.4.185): icmp_seq=3 ttl=234 time=229 ms
64 bytes from redirect-ns.cisco.com (72.163.4.185): icmp_seq=4 ttl=234 time=219 ms
64 bytes from redirect-ns.cisco.com (72.163.4.185): icmp_seq=5 ttl=234 time=232 ms

── cisco.com ping statistics ──
5 packets transmitted, 5 received, 0% packet loss, time 4002ms
rtt min/avg/max/mdev = 218.697/231.968/241.970/7.969 ms

  ┌──(kali㉿kali)-[~]
  └─$ ping -4 localhost
PING localhost (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.020 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.027 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.029 ms
64 bytes from localhost (127.0.0.1): icmp_seq=4 ttl=64 time=0.036 ms
64 bytes from localhost (127.0.0.1): icmp_seq=5 ttl=64 time=0.032 ms
64 bytes from localhost (127.0.0.1): icmp_seq=6 ttl=64 time=0.033 ms
64 bytes from localhost (127.0.0.1): icmp_seq=7 ttl=64 time=0.030 ms
64 bytes from localhost (127.0.0.1): icmp_seq=8 ttl=64 time=0.025 ms
^C
── localhost ping statistics ──
8 packets transmitted, 8 received, 0% packet loss, time 7152ms
rtt min/avg/max/mdev = 0.020/0.029/0.036/0.004 ms

  ┌──(kali㉿kali)-[~]
  └─$ ping -6 localhost
PING localhost (::1) 56 data bytes
64 bytes from localhost (::1): icmp_seq=1 ttl=64 time=0.021 ms
64 bytes from localhost (::1): icmp_seq=2 ttl=64 time=0.028 ms
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.047 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.027 ms
^C
── localhost ping statistics ──
4 packets transmitted, 4 received, 0% packet loss, time 3061ms
rtt min/avg/max/mdev = 0.021/0.030/0.047/0.009 ms

  ┌──(kali㉿kali)-[~]
  └─$ 
```

Player ▾

1 2 3 4

10:55

kali@kali: ~

File Actions Edit View Help

```
  ┌──(kali㉿kali)-[~]
  └─$ ping -6 localhost
PING localhost (::1) 56 data bytes
64 bytes from localhost (::1): icmp_seq=1 ttl=64 time=0.021 ms
64 bytes from localhost (::1): icmp_seq=2 ttl=64 time=0.028 ms
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.047 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.027 ms
^C
── localhost ping statistics ──
4 packets transmitted, 4 received, 0% packet loss, time 3061ms
rtt min/avg/max/mdev = 0.021/0.030/0.047/0.009 ms

  ┌──(kali㉿kali)-[~]
  └─$ ping -M dont localhost -4 -c 5
PING localhost (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.020 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.026 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.026 ms
64 bytes from localhost (127.0.0.1): icmp_seq=4 ttl=64 time=0.026 ms
64 bytes from localhost (127.0.0.1): icmp_seq=5 ttl=64 time=0.041 ms

── localhost ping statistics ──
5 packets transmitted, 5 received, 0% packet loss, time 4093ms
rtt min/avg/max/mdev = 0.020/0.027/0.041/0.007 ms

  ┌──(kali㉿kali)-[~]
  └─$ ping -4n -c20 127.0.0.1 -i 0.2
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.024 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.041 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.026 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.031 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.038 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.031 ms
64 bytes from 127.0.0.1: icmp_seq=11 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=12 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=13 ttl=64 time=0.025 ms
64 bytes from 127.0.0.1: icmp_seq=14 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=15 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=16 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=17 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=18 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=19 ttl=64 time=0.026 ms
64 bytes from 127.0.0.1: icmp_seq=20 ttl=64 time=0.026 ms

── 127.0.0.1 ping statistics ──
20 packets transmitted, 20 received, 0% packet loss, time 3874ms
rtt min/avg/max/mdev = 0.024/0.029/0.041/0.004 ms

  ┌──(kali㉿kali)-[~]
  └─$ 
```

Player ▾

1 2 3 4

11:02

root@kali: /home/kali

File Actions Edit View Help

```
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.031 ms
64 bytes from 127.0.0.1: icmp_seq=11 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=12 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=13 ttl=64 time=0.025 ms
64 bytes from 127.0.0.1: icmp_seq=14 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=15 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=16 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=17 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=18 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=19 ttl=64 time=0.026 ms
64 bytes from 127.0.0.1: icmp_seq=20 ttl=64 time=0.026 ms

—— 127.0.0.1 ping statistics ——
20 packets transmitted, 20 received, 0% packet loss, time 3874ms
rtt min/avg/max/mdev = 0.024/0.029/0.041/0.004 ms

┌──(kali㉿kali)-[~]
└─$ sudo su
[sudo] password for kali:
┌──(root㉿kali)-[/home/kali]
└─# ping -4n -c30 192.168.1.1 -f
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
..............................
—— 192.168.1.1 ping statistics ——
30 packets transmitted, 0 received, 100% packet loss, time 469ms


┌──(root㉿kali)-[/home/kali]
└─# ping -4n -c30 192.168.1.1 -f
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
..............................
—— 192.168.1.1 ping statistics ——
30 packets transmitted, 0 received, 100% packet loss, time 460ms


┌──(root㉿kali)-[/home/kali]
└─# ping -4n -c30 192.168.1.1 -f -i 0.050
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
..............................
—— 192.168.1.1 ping statistics ——
30 packets transmitted, 0 received, 100% packet loss, time 1629ms


┌──(root㉿kali)-[/home/kali]
└─# ping -4 -n -c 30 -f -i 0.050 192.168.1.1

PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
..............................
—— 192.168.1.1 ping statistics ——
30 packets transmitted, 0 received, 100% packet loss, time 1625ms


┌──(root㉿kali)-[/home/kali]
└─#
```

RESULT AND EXAMPLES OF USING PING COMMAND.

We begin by pinging google.com. We open our terminal and type the following [ping google.com]. The ping command will continue to send ICMP packages to the destined IP address until it receives an interruption. To stop the command, just hit the Ctrl + C key combination. As you will see, a number of lines of information will appear on our screen. This shows the packets being sent from our machine to google.com, as well as the response being received. We sent out 14 packets and received 14 packets back, indicating that google.com is up and responding to requests.

  The hostname we are pinging. Use "-n" with this command if you want to avoid any reverse DNS lookups. For example: "ping google.com -n"

 We can see theThe IP address of the target host on the second row on the third column.

On the third column we can see The reverse DNS name of the target IP address. It's different from the original hostname, right? This happens when one hostname has many IP addresses and each IP address has only one DNS name.

On the first column that says 64 bytes The number of data bytes. The default is 56, which translates into 64 ICMP data bytes.

On the fifth column we can see the ICMP sequence numbers for each packet.

On the sixth column we can see TTL: The Time to Live values.

On the seventh column we can see the ping time, measured in milliseconds which is the round trip time for the packet to reach the host, and the response to return to the sender. Greater values indicate possible network problems or target's load.

Once the command stops, it displays a statistic including the percentage of packet loss. The packet loss means that the data was dropped somewhere in the network, indicating an issue within the network or target's performance. If there is a packet loss, you can use the traceroute command to identify where the packet loss occurs.

RTT (Round-trip time) metrics of those ping packages. RTT is the duration in milliseconds it takes for a network request to go from a starting point to a target and back again to the starting point on the last line of the first ping.

We can also use ping to set the packet size using the following commands:

[ping -s 100 localhost]

[ping -s 100 google.com]

Some targets respond to ping packets as expected (1), some of them just drop (2). This is useful when testing a system to see how it will respond differently to very small or very large packets. The default packet size of ping is 56.

As aforementioned, by default, ping will continue to send packages until it receives an interrupt signal. To specify the number of echo request packages to be sent after pings exit, use the -c option followed by the number of packages: [ping -c 5 cisco.com] we can see the result in pic2.

When we run the ping command, it will use either IPv4 or IPv6, depending on your machine's DNS settings. To force ping to use IPv4, pass the -4 option, or use its alias: ping4. To force ping to use IPv6, pass the -6 option, or use its alias: ping6; [ping -4 localhost], [ping -6 localhost]. To send 5 packets which "will not fragment the flag (IPv4 only)" pass "-M dont" option with the following command: [ping -M dont localhost -4 -c 5]

In some cases, it may be necessary to wait a certain amount of time between sending each packet. The default is to wait about one second between each packet, or not to wait in flood mode. Unprivileged users may set an interval to 0.2 seconds and above.

Send 20 ping packages within 0.2 ms interval to target system:[ping -4n -c20 127.0.0.1 -i 0.2]

In flood ping; for every ECHO REQUEST sent a period "." is printed, while for every ECHO REPLY received, the last printed period "." is removed. This provides a rapid display of how many packets are being dropped. If interval is not given, it sets interval to zero and outputs packets as fast as they come back or one hundred times per second, whichever is more. Only the super-user may use this option with a zero interval.

As a root user, flood target system with sending 30 ping packages. Choose your local router or Access Point as the target system. Run this command first [ping -4n -c30 192.168.1.1 -f] then [ping -4n -c30 192.168.1.1 -f -i 0.050]. In this flood test, packet loss is 100% in test number (1), while in test number (2), when the packet is sent with 50 ms

delay, the loss is around 100%. This is because we used an inactive ip, if we used an active ip we should 70% and 30% on the first and second scan respectively.

This is because this feature can be used to slow down the target system's network or to measure end-to-end network performance. It can also be used to create artificial loads. For this reason, it is necessary to be careful when using it on systems in production.

These are a few of the things we can use the PING command for.