PING AND ITS VARIOUS USES

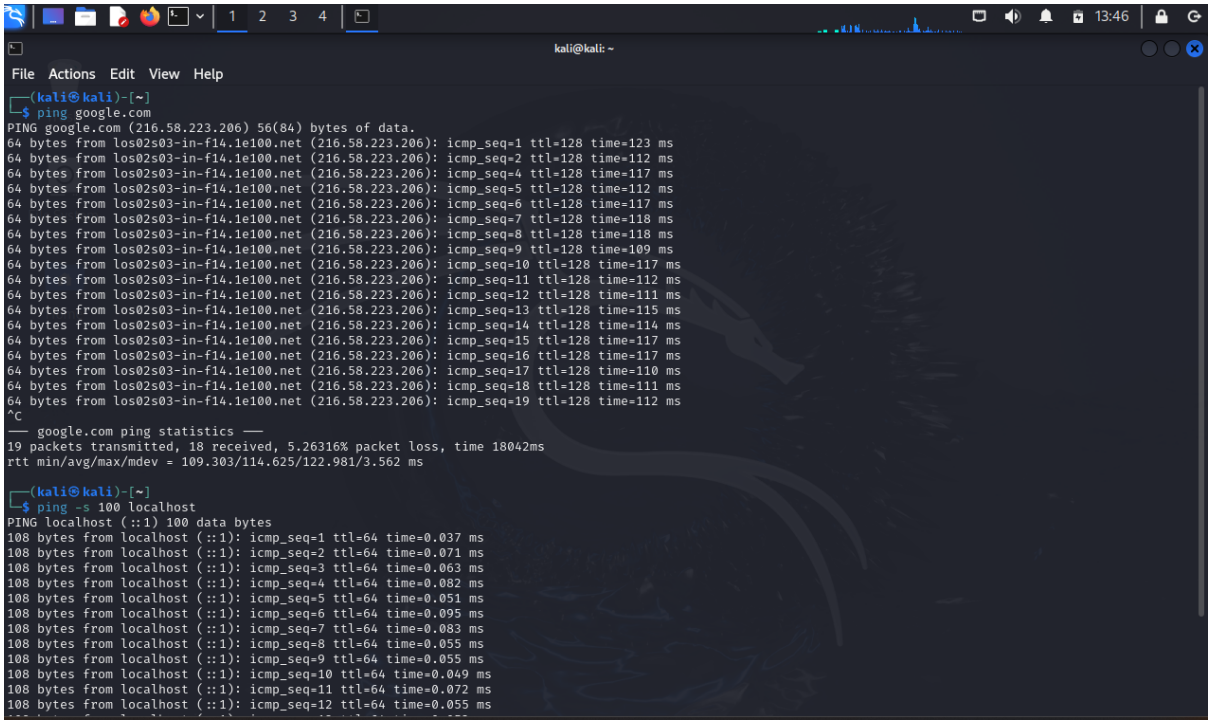
TOOL: KALI LINUX

The `ping` command is a network-based utility used to test if a host (such as a website or server) on an IP network is dead or alive. It also measures the round-trip time for messages sent from the originating host to a destination. `ping` works by sending ICMP (Internet Control Message Protocol) Echo Request messages and waiting for an Echo Reply.

Here are some commonly used parameters with `ping` and their meanings:

Basic Syntax:

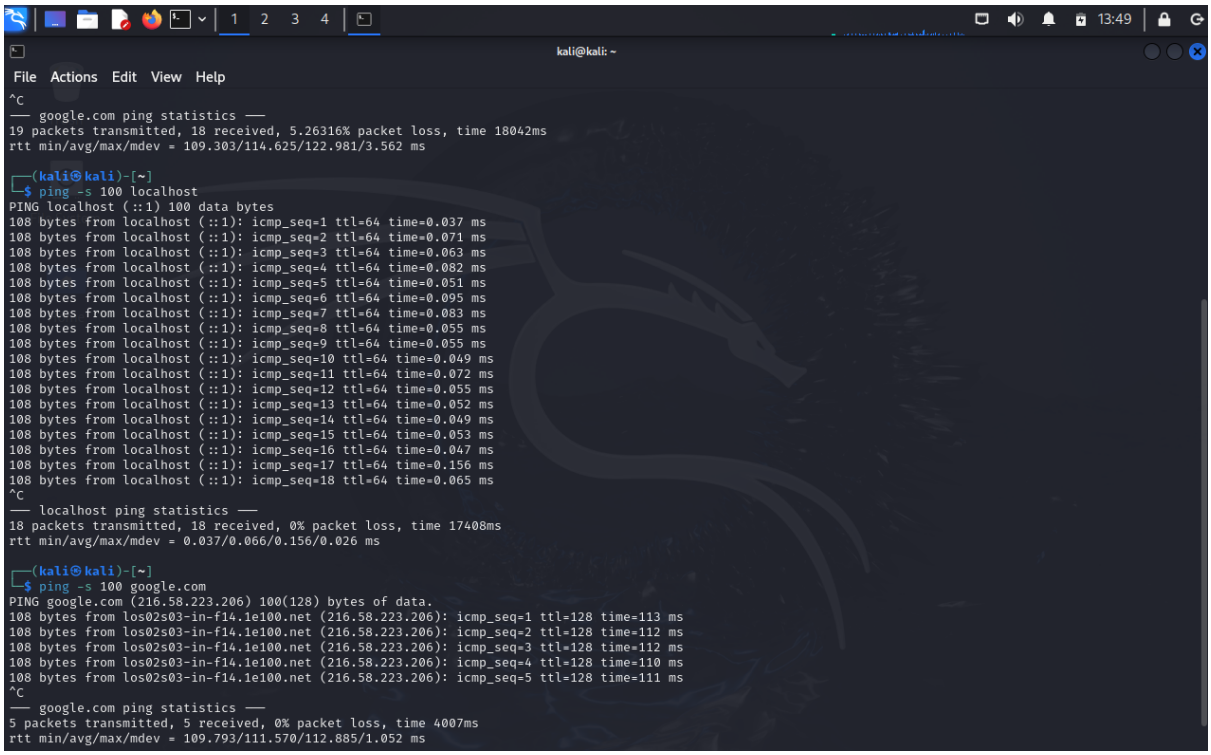
```
ping [options] destination
```



Common Parameters for `ping`:

1. **`-c <count>`**
   - Sends a specific number of packets.
   - Example: `ping -c 4 google.com` — Sends 4 packets to `google.com` and then stops.

2. **`-i <interval>`**
    - Sets the interval in seconds between sending each packet.
    - Example: `ping -i 2 google.com` — Sends a ping every 2 seconds.
3. **`-t <ttl>`**
    - Sets the Time-to-Live (TTL) value, which is the maximum number of hops (routers) the packet can take before being discarded. It helps identify how many routers a packet passes through.
    - Example: `ping -t 10 google.com` — Sets the TTL to 10.
4. **`-s <size>`**
    - Specifies the size of the packet in bytes.
    - Example: `ping -s 1000 google.com` — Sends packets of size 1000 bytes.



5. **`-q`**
    - Suppresses output except for the summary (useful for script automation).
    - Example: `ping -q google.com` — Only shows the summary of the ping results.
6. **`-n`**

    - In some versions of `ping`, this option forces numeric output (i.e., IP addresses instead of hostnames). This may be particularly useful if DNS resolution is slow or unreliable.
    - Example: `ping -n 1 8.8.8.8` — Sends one ping to the IP address `8.8.8.8`.
7. **`-a`**
    - Audibly alerts the user for each reply (on some systems, mostly Linux).
    - Example: `ping -a google.com` — Plays a sound for every received reply.
8. **`-l <size>`**
    - (Windows-specific) Specifies the send buffer size in bytes.
    - Example: `ping -l 1024 google.com` — Sends 1024-byte packets.
9. **`-W <timeout>`**
    - Specifies the timeout in seconds to wait for a response.
    - Example: `ping -W 3 google.com` — Waits for 3 seconds before timing out.
10. **`-v`**
    - In some versions, this enables verbose output, providing more detailed information about each packet.
    - Example: `ping -v google.com` — Prints detailed information about each packet's trip.
11. **`-R`**
    - (Linux-specific) Traces the route back to the source, showing how the packets are routed through various hops.
    - Example: `ping -R google.com` — Displays the route back to the source.
12. **`-p <pattern>`**
    - (Linux-specific) Sends a custom pattern of bytes in the ICMP packet.
    - Example: `ping -p ff google.com` — Sends packets with the `ff` byte pattern.

Example Commands:

- **Send 5 pings to a host**

  ```
  ping -c 5 google.com
  ```

- **Set a packet size of 1200 bytes and 10-second interval:**

```
ping -s 1200 -i 10 google.com
```

- **Check if a host is reachable with minimal output:**

```
ping -q google.com
```

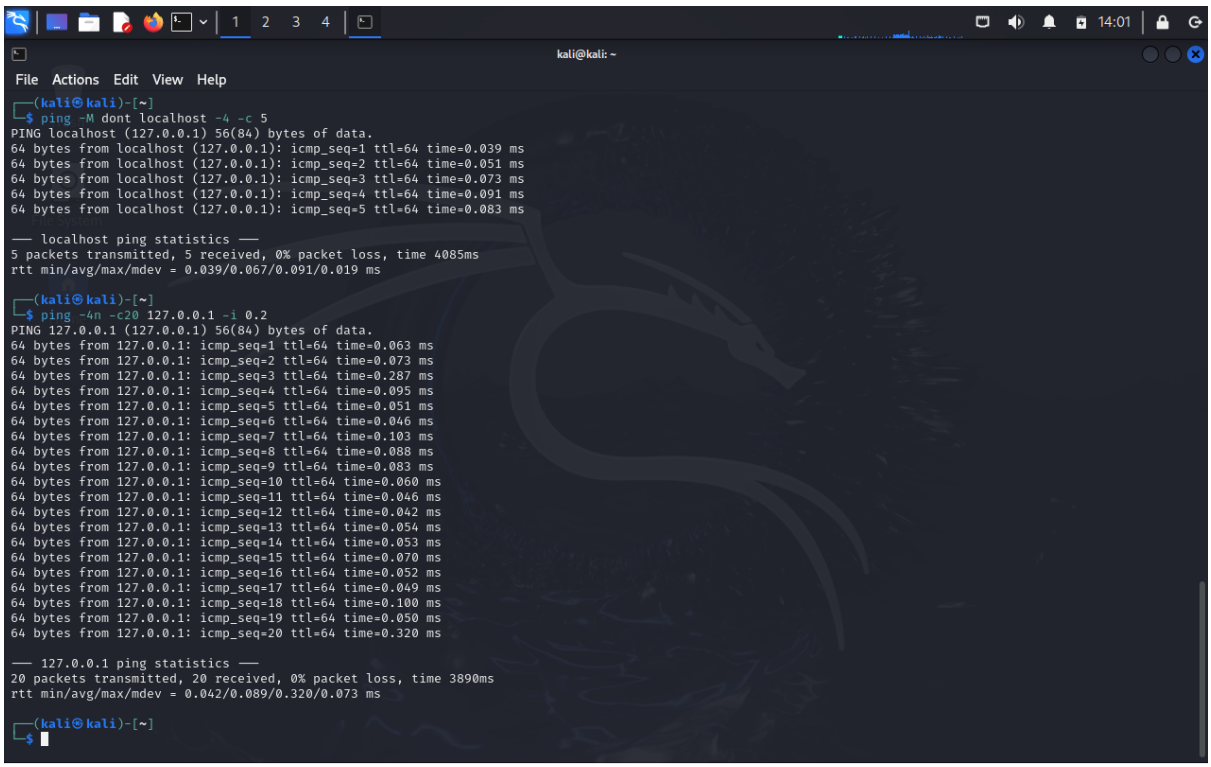
- **Test a server with a specific TTL value:**

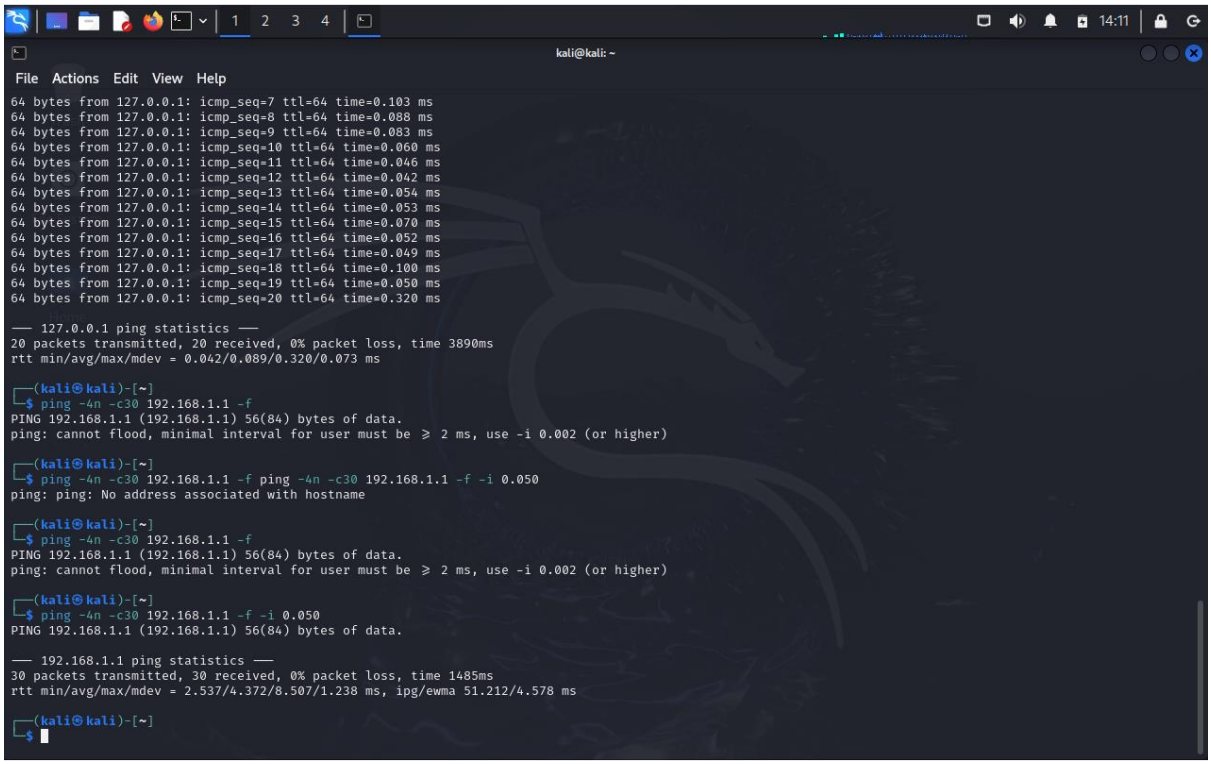
```
ping -t 64 google.com
```

- **Ping with a timeout of 2 seconds:**

```
ping -W 2 google.com
```

These are a few examples of the various parameters to use with `ping`. Depending on the operating system (Linux, macOS, Windows).




1 2 3 4
14:11
kali@kali: ~
File Actions Edit View Help
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.103 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.088 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.083 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.060 ms
64 bytes from 127.0.0.1: icmp_seq=11 ttl=64 time=0.046 ms
64 bytes from 127.0.0.1: icmp_seq=12 ttl=64 time=0.042 ms
64 bytes from 127.0.0.1: icmp_seq=13 ttl=64 time=0.054 ms
64 bytes from 127.0.0.1: icmp_seq=14 ttl=64 time=0.053 ms
64 bytes from 127.0.0.1: icmp_seq=15 ttl=64 time=0.070 ms
64 bytes from 127.0.0.1: icmp_seq=16 ttl=64 time=0.052 ms
64 bytes from 127.0.0.1: icmp_seq=17 ttl=64 time=0.049 ms
64 bytes from 127.0.0.1: icmp_seq=18 ttl=64 time=0.100 ms
64 bytes from 127.0.0.1: icmp_seq=19 ttl=64 time=0.050 ms
64 bytes from 127.0.0.1: icmp_seq=20 ttl=64 time=0.320 ms
— 127.0.0.1 ping statistics —
20 packets transmitted, 20 received, 0% packet loss, time 3890ms
rtt min/avg/max/mdev = 0.042/0.089/0.320/0.073 ms
┌──(kali㉿kali)-[~]
└─$ ping -4n -c30 192.168.1.1 -f
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
ping: cannot flood, minimal interval for user must be ≥ 2 ms, use -i 0.002 (or higher)
┌──(kali㉿kali)-[~]
└─$ ping -4n -c30 192.168.1.1 -f ping -4n -c30 192.168.1.1 -f -i 0.050
ping: ping: No address associated with hostname
┌──(kali㉿kali)-[~]
└─$ ping -4n -c30 192.168.1.1 -f
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
ping: cannot flood, minimal interval for user must be ≥ 2 ms, use -i 0.002 (or higher)
┌──(kali㉿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, 30 received, 0% packet loss, time 1485ms
rtt min/avg/max/mdev = 2.537/4.372/8.507/1.238 ms, ipg/ewma 51.212/4.578 ms
┌──(kali㉿kali)-[~]
└─$