

NETWORK DEVICE DISCOVERY WITH ARP



YOUTUBE.COM/@INFORMATIONTECHWITHDAVE



Can't this knowledge be used by 3v1l h4XX0rz?

- YES. Unauthorized hackers can use arp tools to discover hosts to target on a network.
- Some of these tools are installed by default on systems, so hackers that like to “live off the land,” especially APTs (Advanced Persistent Threats) will rely on them.
- So, network administrators and Blue Team operators need to understand them, and can also use them to discover unauthorized hosts connected to their networks.
- Sun Tzu noted in The Art of War:
 - “If you know the enemy and know yourself, you need not fear the result of a hundred battles. If you know yourself but not the enemy, for every victory gained you will also suffer a defeat. If you know neither the enemy nor yourself, you will succumb in every battle.”
- As with any tool, intent matters!

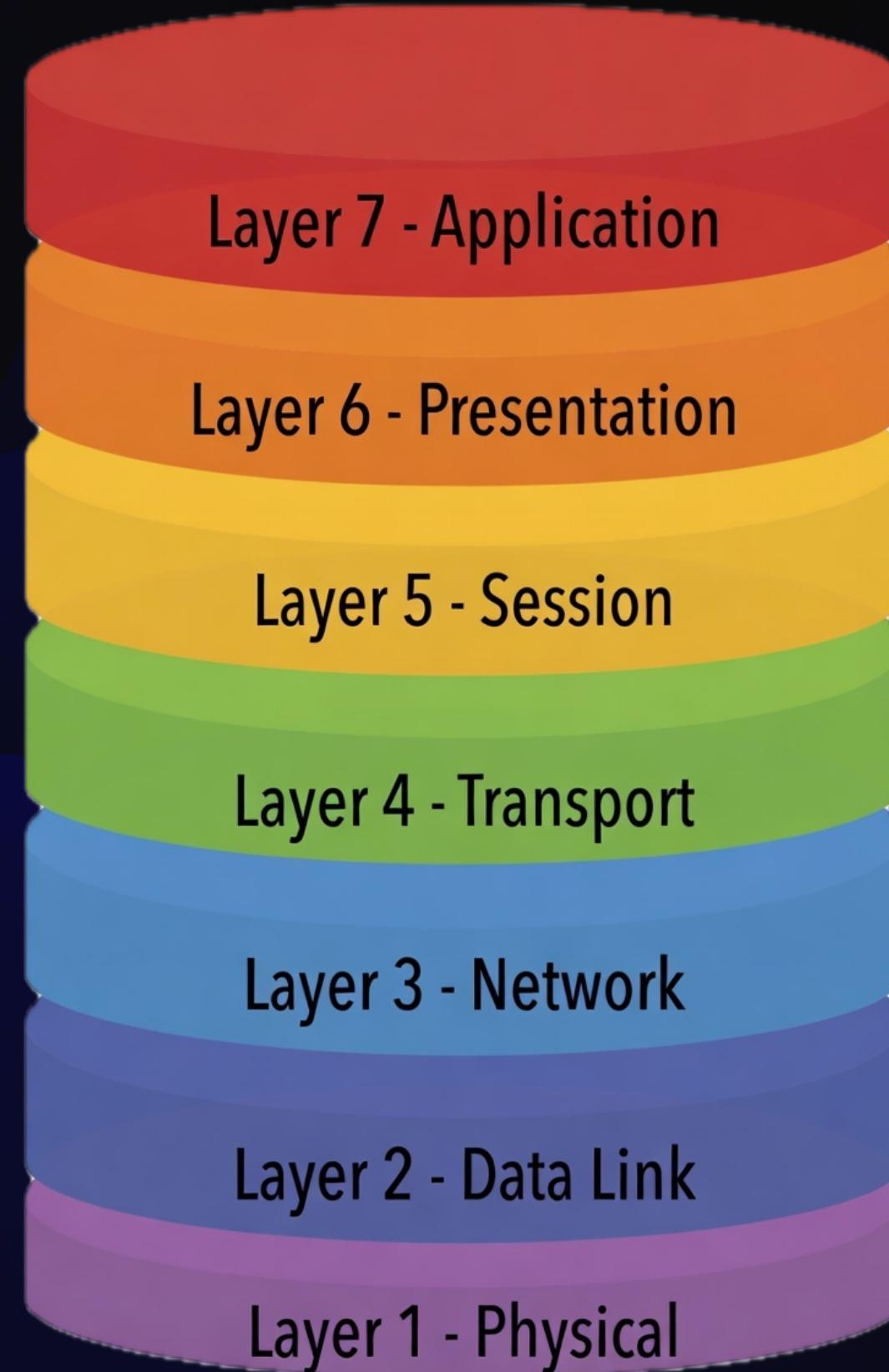


We're UNIX-Focused

- This presentation will be focused on UNIX and UNIX-like operating systems on servers and workstations, but be aware that network devices like routers and switches also rely on ARP and have their own implementations.
- We're focusing on Linux, macOS, and BSD-based systems.
- If you're on Windows, check out the Windows Subsystem for Linux, or install a Linux virtual machine in VirtualBox.



ARP - Address Resolution Protocol



- ARP is a networking protocol that works at Layer 2 of the OSI Model.
- Built-in on all network-capable operating systems.
- Maps physical (MAC) addresses to IPv4 addresses.
- NOTE: IPv6 uses other protocols like Neighbor Discovery Protocol and Neighbor Advertisement Messages instead of ARP.
- Since it works at Layer 2, ARP resolution is confined to a single broadcast domain. I.e., subnets and VLANs.
- ARP packets do not traverse routers.



How ARP Maps a Physical Address to an IP

- Before diving into how to scan with ARP we need to understand how it works.
- It's a simple two-step process.
- Step 1: Host A sends an ARP broadcast request for the physical address associated with an IPv4 address.
- Step 2: Host B with that IP replies with its MAC address.

The Address Resolution Protocol (ARP) works strictly within the local subnet (Layer 2). An ARP request is a broadcast, which ***must*** be processed and answered by any active, configured device, regardless of Layer 3 firewall rules.

1. Host A Broadcasts

"Who has IP 192.168.1.10?"

Firewall (Ignores) Other Subnet (Ignores)

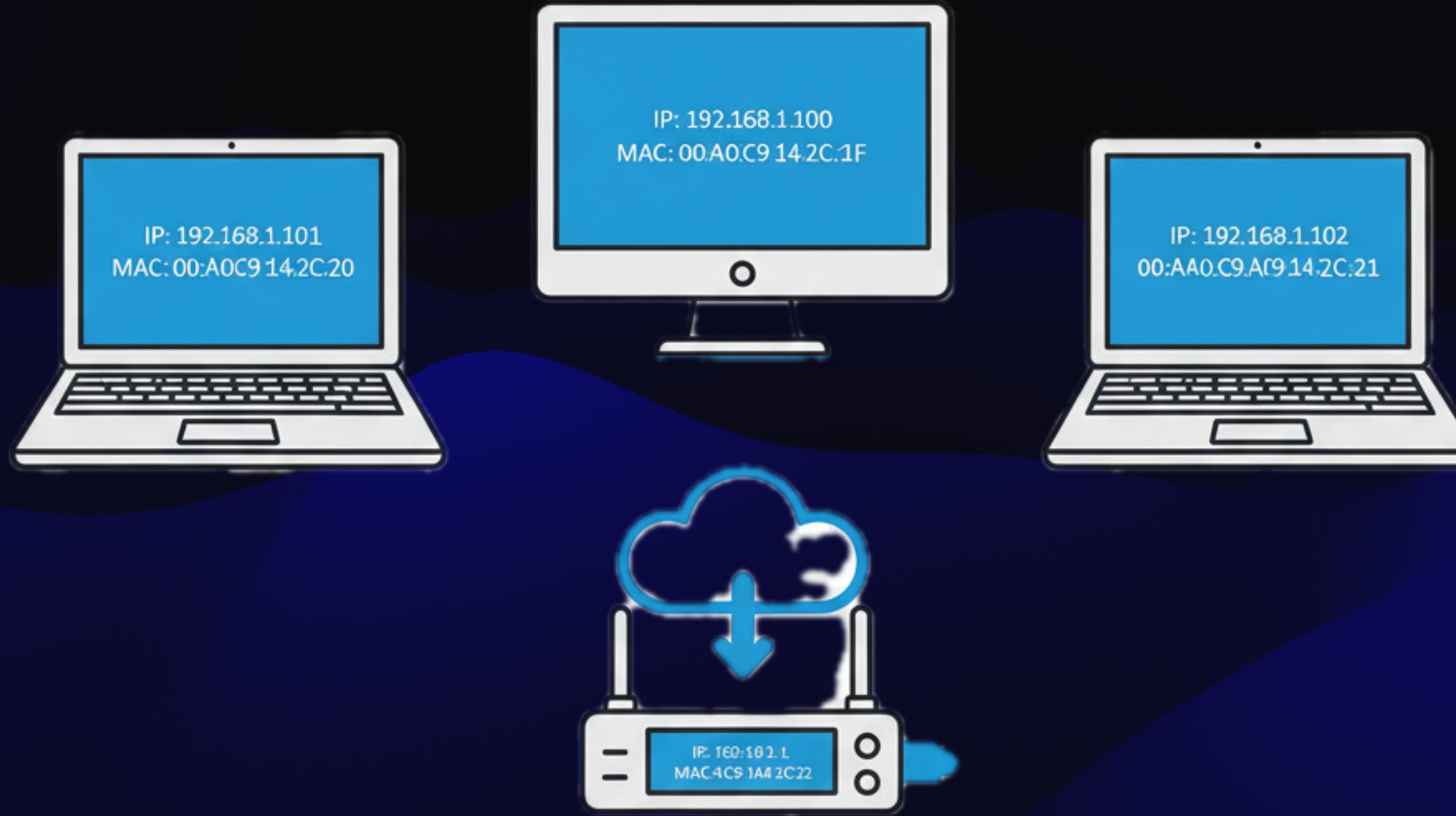
2. Target Host B Replies (Unicast)

"I do! My MAC is 00:1A:2B:3C:4D:5E."

Result: IP mapped to MAC, bypassing L3 restrictions.



Why ARP is useful for device discovery



- Devices must respond to ARP to communicate on a LAN over IPv4.
- Many devices are configured to ignore ping requests. But even devices that don't respond to ping (ICMP packets) will respond to ARP queries.
- Local host firewalls that block all TCP, UDP, and ICMP packets will generally not block ARP.



Tools for Device Discovery with ARP

- arp - System command part of net-tools suite (deprecated on Linux but still used on macOS)
- ip neighbor - System command part of iproute2 suite
- arp-scan - <https://github.com/royhills/arp-scan>
- nmap - <https://nmap.org/>
- netdiscover - <https://github.com/netdiscover-scanner/netdiscover>
- tcpdump - <https://www.tcpdump.org/>
- oui.is - Look up device manufacturer based on MAC address



arp command

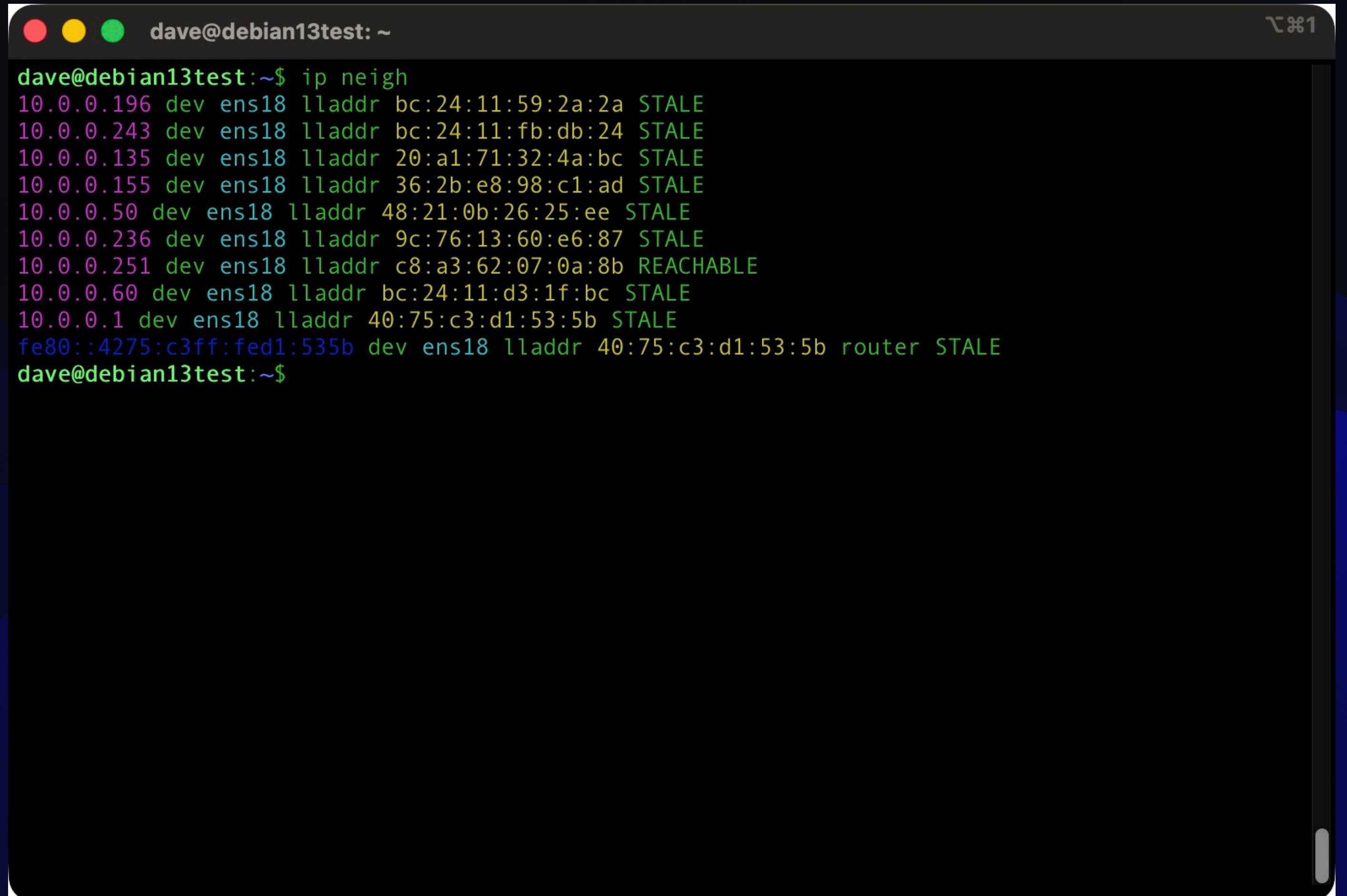
```
-bash
Daves-M4-MacBook-Air:~ dave$ arp -i en0 -a
? (10.0.0.1) at 40:75:c3:d1:53:5b on en0 ifs scope [ethernet]
? (10.0.0.114) at 40:d1:60:bb:4:fa on en0 ifs scope [ethernet]
? (10.0.0.116) at ce:b2:cc:90:9:25 on en0 ifs scope [ethernet]
? (10.0.0.132) at 68:ef:dc:88:f2:ef on en0 ifs scope [ethernet]
? (10.0.0.136) at ce:e5:57:4f:db:ea on en0 ifs scope [ethernet]
? (10.0.0.140) at 0:1b:a9:b8:db:f6 on en0 ifs scope [ethernet]
? (10.0.0.141) at c2:d5:53:64:ae:47 on en0 ifs scope [ethernet]
? (10.0.0.142) at ce:3e:83:98:5b:8f on en0 ifs scope [ethernet]
? (10.0.0.153) at dc:9e:8f:f1:95:ba on en0 ifs scope [ethernet]
? (10.0.0.165) at ce:78:67:2c:dc:b1 on en0 ifs scope [ethernet]
? (10.0.0.185) at f4:39:a6:b:c2:8d on en0 ifs scope [ethernet]
? (10.0.0.193) at 76:b4:e3:3d:3a:84 on en0 ifs scope [ethernet]
? (10.0.0.221) at ce:7:85:80:25:f8 on en0 ifs scope [ethernet]
? (10.0.0.230) at 6a:9e:56:18:1a:54 on en0 ifs scope permanent [ethernet]
? (10.0.0.236) at 9c:76:13:60:e6:87 on en0 ifs scope [ethernet]
? (10.0.0.243) at bc:24:11:fb:db:24 on en0 ifs scope [ethernet]
? (10.0.0.246) at 6a:7c:ed:36:79:2 on en0 ifs scope [ethernet]
? (10.0.0.249) at 9c:76:13:60:e3:85 on en0 ifs scope [ethernet]
? (10.0.0.251) at c8:a3:62:7:a:8b on en0 ifs scope [ethernet]
mdns.mcast.net (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifs scope permanent [ethernet]
Daves-M4-MacBook-Air:~ dave$
```

- Deprecated on Linux but still used on macOS.
- Quick and easy way to display your machines arp cache.
- \$ arp -i en0 -a
- Shows host IPs, MACs, and which local interface they were seen on.
- If you don't specify your machine's interface (e.g., en0) it will use all of them, so you may get duplicates.



ip neighbor / ip neigh command on Linux

- Linux still supports arp but that is being deprecated, and it may not be installed by default.
- ip neighbor, or ip neigh for short, is the newer way to display the system's arp cache.
- Displays host IPs, which local interface they are seen on, MAC, and status of the cache entry.



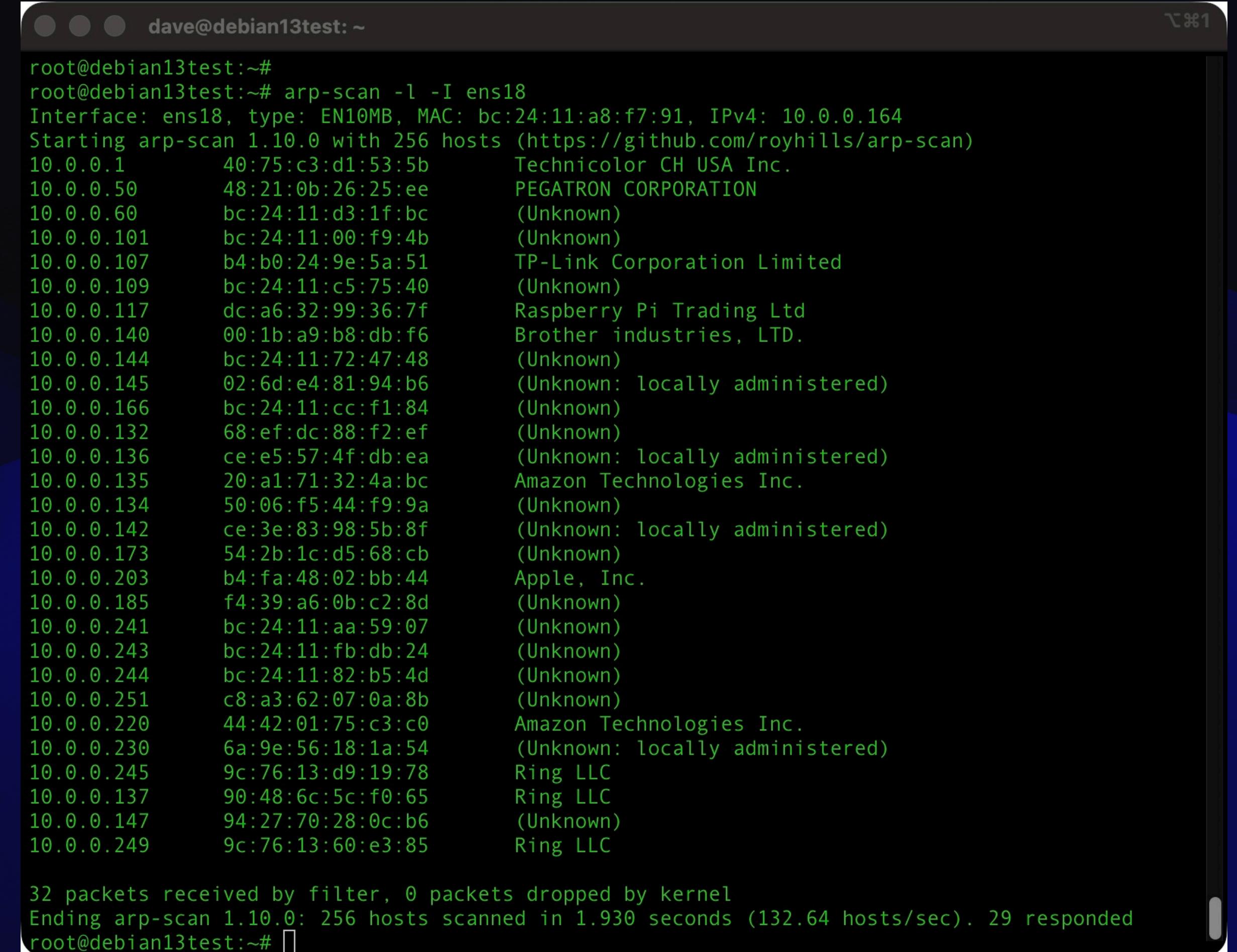
The screenshot shows a terminal window with a dark background and light-colored text. At the top, there are three colored circles (red, yellow, green) followed by the text "dave@debian13test: ~". In the bottom right corner of the terminal window, there is a small icon of a penguin wearing a cap. The main content of the terminal is the output of the "ip neigh" command, which lists various network neighbors with their IP addresses, local interfaces, MAC addresses, and status (e.g., STALE, REACHABLE). The output is as follows:

```
dave@debian13test:~$ ip neigh
10.0.0.196 dev ens18 lladdr bc:24:11:59:2a:2a STALE
10.0.0.243 dev ens18 lladdr bc:24:11:fb:db:24 STALE
10.0.0.135 dev ens18 lladdr 20:a1:71:32:4a:bc STALE
10.0.0.155 dev ens18 lladdr 36:2b:e8:98:c1:ad STALE
10.0.0.50 dev ens18 lladdr 48:21:0b:26:25:ee STALE
10.0.0.236 dev ens18 lladdr 9c:76:13:60:e6:87 STALE
10.0.0.251 dev ens18 lladdr c8:a3:62:07:0a:8b REACHABLE
10.0.0.60 dev ens18 lladdr bc:24:11:d3:1f:bc STALE
10.0.0.1 dev ens18 lladdr 40:75:c3:d1:53:5b STALE
fe80::4275:c3ff:fed1:535b dev ens18 lladdr 40:75:c3:d1:53:5b router STALE
dave@debian13test:~$
```



Host Discovery Using arp-scan (1 of 2)

- arp-scan discovers and fingerprints hosts using ARP.
- Runs on Linux, BSD, and macOS (install on Mac with Homebrew).
- Like the other tools, results may vary depending on what switch port you're connected to.
- On Linux, you may also need to run as root rather than using sudo, to avoid permissions issues.



```
dave@debian13test:~
```

```
root@debian13test:~#
root@debian13test:~# arp-scan -l -I ens18
Interface: ens18, type: EN10MB, MAC: bc:24:11:a8:f7:91, IPv4: 10.0.0.164
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
10.0.0.1      40:75:c3:d1:53:5b      Technicolor CH USA Inc.
10.0.0.50     48:21:0b:26:25:ee      PEGATRON CORPORATION
10.0.0.60     bc:24:11:d3:1f:bc      (Unknown)
10.0.0.101    bc:24:11:00:f9:4b      (Unknown)
10.0.0.107    b4:b0:24:9e:5a:51      TP-Link Corporation Limited
10.0.0.109    bc:24:11:c5:75:40      (Unknown)
10.0.0.117    dc:a6:32:99:36:7f      Raspberry Pi Trading Ltd
10.0.0.140    00:1b:a9:b8:db:f6      Brother industries, LTD.
10.0.0.144    bc:24:11:72:47:48      (Unknown)
10.0.0.145    02:6d:e4:81:94:b6      (Unknown: locally administered)
10.0.0.166    bc:24:11:cc:f1:84      (Unknown)
10.0.0.132    68:ef:dc:88:f2:ef      (Unknown)
10.0.0.136    ce:e5:57:4f:db:ea      (Unknown: locally administered)
10.0.0.135    20:a1:71:32:4a:bc      Amazon Technologies Inc.
10.0.0.134    50:06:f5:44:f9:9a      (Unknown)
10.0.0.142    ce:3e:83:98:5b:8f      (Unknown: locally administered)
10.0.0.173    54:2b:1c:d5:68:cb      (Unknown)
10.0.0.203    b4:fa:48:02:bb:44      Apple, Inc.
10.0.0.185    f4:39:a6:0b:c2:8d      (Unknown)
10.0.0.241    bc:24:11:aa:59:07      (Unknown)
10.0.0.243    bc:24:11:fb:db:24      (Unknown)
10.0.0.244    bc:24:11:82:b5:4d      (Unknown)
10.0.0.251    c8:a3:62:07:0a:8b      (Unknown)
10.0.0.220    44:42:01:75:c3:c0      Amazon Technologies Inc.
10.0.0.230    6a:9e:56:18:1a:54      (Unknown: locally administered)
10.0.0.245    9c:76:13:d9:19:78      Ring LLC
10.0.0.137    90:48:6c:5c:f0:65      Ring LLC
10.0.0.147    94:27:70:28:0c:b6      (Unknown)
10.0.0.249    9c:76:13:60:e3:85      Ring LLC

32 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.930 seconds (132.64 hosts/sec). 29 responded
root@debian13test:~#
```



Host Discovery Using arp-scan (2 of 2)

```
-bash
Daves-M4-MacBook-Air:~ dave$ sudo arp-scan -I en0 -l
Interface: en0, type: EN10MB, MAC: 6a:9e:56:18:1a:54, IPv4: 10.0.0.230
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
10.0.0.1      40:75:c3:d1:53:5b      Technicolor CH USA Inc.
10.0.0.50     48:21:0b:26:25:ee      PEGATRON CORPORATION
10.0.0.60      bc:24:11:d3:1f:bc      (Unknown)
10.0.0.101     bc:24:11:00:f9:4b      (Unknown)
10.0.0.107     b4:b0:24:9e:5a:51      TP-Link Corporation Limited
10.0.0.109     bc:24:11:c5:75:40      (Unknown)
10.0.0.117     dc:a6:32:99:36:7f      Raspberry Pi Trading Ltd
10.0.0.140     00:1b:a9:b8:db:f6      Brother industries, LTD.
10.0.0.144     bc:24:11:72:47:48      (Unknown)
10.0.0.145     02:6d:e4:81:94:b6      (Unknown: locally administered)
10.0.0.164     bc:24:11:a8:f7:91      (Unknown)
10.0.0.166     bc:24:11:cc:f1:84      (Unknown)
10.0.0.137     90:48:6c:5c:f0:65      Ring LLC
10.0.0.134     50:06:f5:44:f9:9a      (Unknown)
10.0.0.135     20:a1:71:32:4a:bc      Amazon Technologies Inc.
10.0.0.136     ce:e5:57:4f:db:ea      (Unknown: locally administered)
10.0.0.142     ce:3e:83:98:5b:8f      (Unknown: locally administered)
10.0.0.132     68:ef:dc:88:f2:ef      (Unknown)
10.0.0.241     bc:24:11:aa:59:07      (Unknown)
10.0.0.147     94:27:70:28:0c:b6      (Unknown)
10.0.0.185     f4:39:a6:0b:c2:8d      (Unknown)
10.0.0.243     bc:24:11:fb:db:24      (Unknown)
10.0.0.244     bc:24:11:82:b5:4d      (Unknown)
10.0.0.251     c8:a3:62:07:0a:8b      (Unknown)
10.0.0.203     b4:fa:48:02:bb:44      Apple, Inc.
10.0.0.245     9c:76:13:d9:19:78      Ring LLC
10.0.0.173     54:2b:1c:d5:68:cb      (Unknown)
10.0.0.220     44:42:01:75:c3:c0      Amazon Technologies Inc.
10.0.0.139     50:f2:65:85:38:33      (Unknown)

519 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.829 seconds (139.97 hosts/sec). 29 responded
Daves-M4-MacBook-Air:~ dave$
```

- Here is the output from my Mac on a different switch port but on the same network as the Linux example on the previous slide.
- Note here that it ran properly when invoked via sudo.
- Output is nicely formatted into columns with the IP, MAC, and vendor (where known).



Nmap arp scanning

- Nmap can do arp scans of a subnet and return a list of connected hosts.
- Check out my other video, [Exploring and Scanning with Nmap](#).

```
Daves-M4-MacBook-Air:~ dave$ sudo nmap -sn -PR 10.0.0.0/24
Password:

Starting Nmap 7.98 ( https://nmap.org ) at 2025-11-15 17:42 -0500
Nmap scan report for 10.0.0.50
Host is up (0.00047s latency).

MAC Address: 48:21:0B:26:25:EE (Pegatron)

Nmap scan report for 10.0.0.60
Host is up (0.00065s latency).

MAC Address: BC:24:11:D3:1F:BC (Proxmox Server Solutions GmbH)

Nmap scan report for 10.0.0.101
Host is up (0.00062s latency).

MAC Address: BC:24:11:00:F9:4B (Proxmox Server Solutions GmbH)

Nmap scan report for 10.0.0.107
Host is up (0.034s latency).

MAC Address: B4:B0:24:9E:5A:51 (TP-Link Limited)

Nmap scan report for 10.0.0.109
Host is up (0.00049s latency).

MAC Address: BC:24:11:C5:75:40 (Proxmox Server Solutions GmbH)

Nmap scan report for 10.0.0.117
Host is up (0.00042s latency).

MAC Address: DC:A6:32:99:36:7F (Raspberry Pi Trading)

Nmap scan report for 10.0.0.134
Host is up (0.19s latency).

MAC Address: 50:06:F5:44:F9:9A (Roku)

(Output deleted for brevity.)
```



netdiscover scanning

```
dave@debian13test:~$  
dave@debian13test:~$ sudo netdiscover -r 10.0.0.0/24  
[sudo] password for dave:  
Currently scanning: Finished! | Screen View: Unique Hosts  
73 Captured ARP Req/Rep packets, from 31 hosts. Total size: 4020  


| IP         | At MAC Address    | Count | Len  | MAC Vendor / Hostname         |
|------------|-------------------|-------|------|-------------------------------|
| 10.0.0.243 | bc:24:11:fb:db:24 | 12    | 504  | Proxmox Server Solutions GmbH |
| 10.0.0.117 | dc:a6:32:99:36:7f | 28    | 1680 | Raspberry Pi Trading Ltd      |
| 10.0.0.1   | 40:75:c3:d1:53:5b | 2     | 120  | Vantiva USA LLC               |
| 10.0.0.50  | 48:21:0b:26:25:ee | 1     | 42   | PEGATRON CORPORATION          |
| 10.0.0.60  | bc:24:11:d3:1f:bc | 1     | 42   | Proxmox Server Solutions GmbH |
| 10.0.0.101 | bc:24:11:00:f9:4b | 1     | 42   | Proxmox Server Solutions GmbH |
| 10.0.0.107 | b4:b0:24:9e:5a:51 | 2     | 120  | TP-Link Systems Inc           |
| 10.0.0.109 | bc:24:11:c5:75:40 | 1     | 42   | Proxmox Server Solutions GmbH |
| 10.0.0.140 | 00:1b:a9:b8:db:f6 | 1     | 60   | Brother industries, LTD.      |
| 10.0.0.144 | bc:24:11:72:47:48 | 1     | 42   | Proxmox Server Solutions GmbH |
| 10.0.0.145 | 02:6d:e4:81:94:b6 | 1     | 42   | Unknown vendor                |
| 10.0.0.166 | bc:24:11:cc:f1:84 | 1     | 42   | Proxmox Server Solutions GmbH |
| 10.0.0.116 | ce:b2:cc:90:09:25 | 1     | 60   | Unknown vendor                |
| 10.0.0.241 | bc:24:11:aa:59:07 | 1     | 60   | Proxmox Server Solutions GmbH |
| 10.0.0.244 | bc:24:11:82:b5:4d | 1     | 42   | Proxmox Server Solutions GmbH |
| 10.0.0.251 | c8:a3:62:07:0a:8b | 1     | 60   | ASIX Electronics Corporation  |
| 10.0.0.134 | 50:06:f5:44:f9:9a | 1     | 60   | Roku, Inc                     |
| 10.0.0.135 | 20:a1:71:32:4a:bc | 1     | 60   | Amazon Technologies Inc.      |
| 10.0.0.136 | ce:e5:57:4f:db:ea | 1     | 60   | Unknown vendor                |
| 10.0.0.142 | ce:3e:83:98:5b:8f | 1     | 60   | Unknown vendor                |
| 10.0.0.147 | 94:27:70:28:0c:b6 | 1     | 60   | BSH Hausgeräte GmbH           |
| 10.0.0.137 | 90:48:6c:5c:f0:65 | 2     | 120  | Ring LLC                      |
| 10.0.0.173 | 54:2b:1c:d5:68:cb | 1     | 60   | Amazon Technologies Inc.      |
| 10.0.0.185 | f4:39:a6:0b:c2:8d | 1     | 60   | Apple, Inc.                   |
| 10.0.0.220 | 44:42:01:75:c3:c0 | 1     | 60   | Amazon Technologies Inc.      |
| 10.0.0.230 | 6a:9e:56:18:1a:54 | 1     | 60   | Unknown vendor                |
| 10.0.0.221 | ce:07:85:80:25:f8 | 1     | 60   | Unknown vendor                |
| 10.0.0.203 | b4:fa:48:02:bb:44 | 1     | 60   | Apple, Inc.                   |
| 10.0.0.245 | 9c:76:13:d9:19:78 | 1     | 60   | Ring LLC                      |
| 10.0.0.249 | 9c:76:13:60:e3:85 | 2     | 120  | Ring LLC                      |
| 10.0.0.236 | 9c:76:13:60:e6:87 | 1     | 60   | Ring LLC                      |

  
dave@debian13test:~$
```

- netdiscover can be used actively or passively.

- Largely intended for use on WiFi networks but also works on wired networks.

- Can work on networks without a DHCP server.

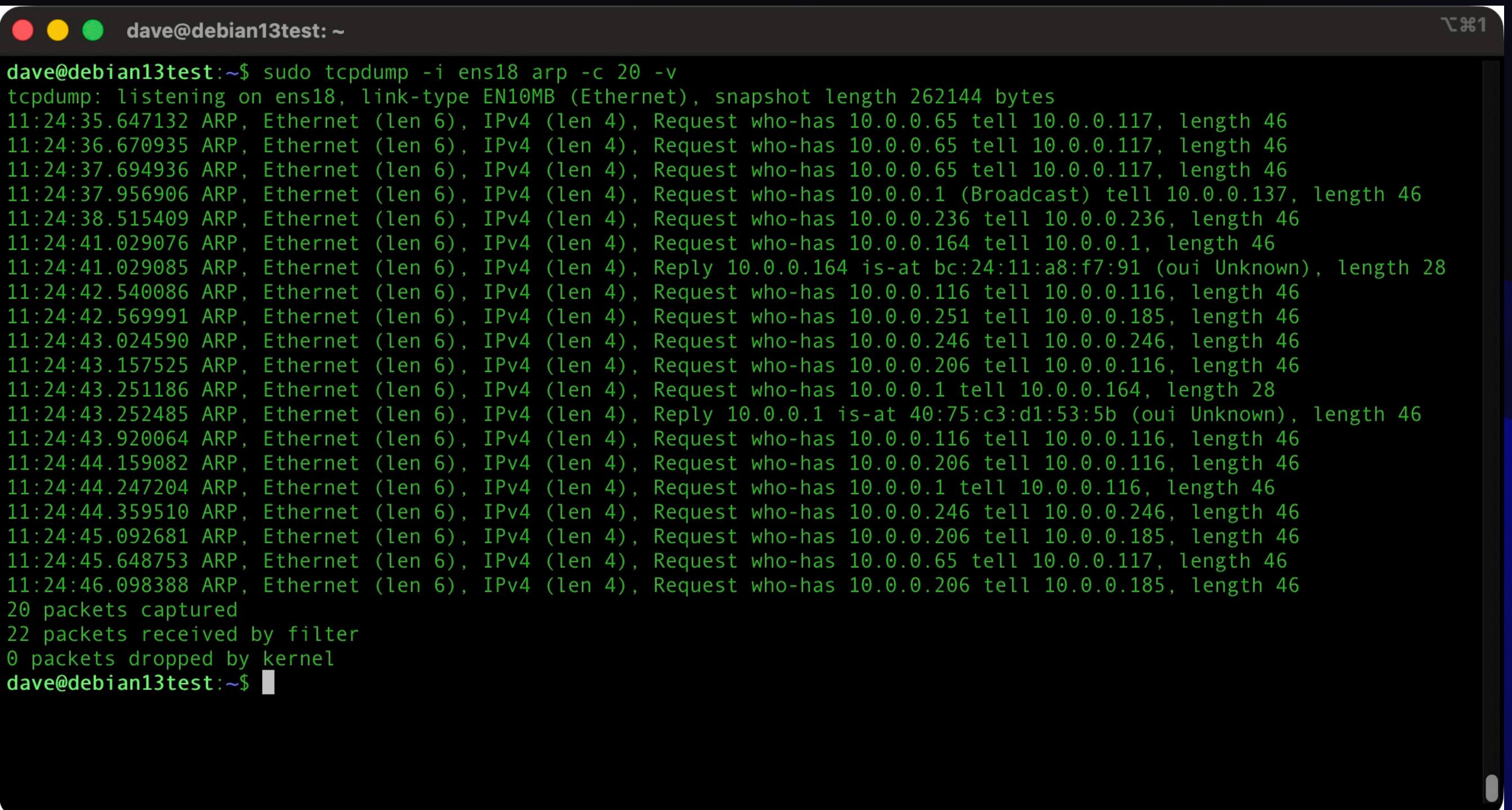
- Use the “-r” option to specify scan range.

- Use the “-i” option to specify your network device if needed.



Using tcpdump to watch the wire

- tcpdump is a powerful commandline tool relied on by many other tools like Wireshark.
 - It can be used on its own for live packet captures.
 - Setup filters to limit captures to only what you're interested in.
 - This example runs tcpdump, listening on interface ens18, only captures arp packets, and exits after 100 packets are captured. “-v” specifies verbose output.
-
- sudo tcpdump -i ens18 arp -c 20
-v



A screenshot of a terminal window titled "dave@debian13test: ~". The window shows the command "sudo tcpdump -i ens18 arp -c 20 -v" being run. The output displays 20 ARP requests and replies on the ens18 interface. The requests are from various MAC addresses (e.g., 11:24:35.647132, 11:24:36.670935) to the broadcast address (11:24:37.956906). The replies are from the MAC address 11:24:41.029076. The output includes detailed information about each packet, such as source and destination MAC and IP addresses, and the length of the payload.

```
dave@debian13test:~$ sudo tcpdump -i ens18 arp -c 20 -v
tcpdump: listening on ens18, link-type EN10MB (Ethernet), snapshot length 262144 bytes
11:24:35.647132 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.65 tell 10.0.0.117, length 46
11:24:36.670935 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.65 tell 10.0.0.117, length 46
11:24:37.694936 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.65 tell 10.0.0.117, length 46
11:24:37.956906 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.1 (Broadcast) tell 10.0.0.137, length 46
11:24:38.515409 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.236 tell 10.0.0.236, length 46
11:24:41.029076 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.164 tell 10.0.0.1, length 46
11:24:41.029085 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.0.0.164 is-at bc:24:11:a8:f7:91 (oui Unknown), length 28
11:24:42.540086 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.116 tell 10.0.0.116, length 46
11:24:42.569991 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.251 tell 10.0.0.185, length 46
11:24:43.024590 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.246 tell 10.0.0.246, length 46
11:24:43.157525 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.206 tell 10.0.0.116, length 46
11:24:43.251186 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.1 tell 10.0.0.164, length 28
11:24:43.252485 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.0.0.1 is-at 40:75:c3:d1:53:5b (oui Unknown), length 46
11:24:43.920064 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.116 tell 10.0.0.116, length 46
11:24:44.159082 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.206 tell 10.0.0.116, length 46
11:24:44.247204 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.1 tell 10.0.0.116, length 46
11:24:44.359510 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.246 tell 10.0.0.246, length 46
11:24:45.092681 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.206 tell 10.0.0.185, length 46
11:24:45.648753 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.65 tell 10.0.0.117, length 46
11:24:46.098388 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 10.0.0.206 tell 10.0.0.185, length 46
20 packets captured
22 packets received by filter
0 packets dropped by kernel
dave@debian13test:~$
```



Search for Device Manufacturers

The screenshot shows a web browser window with the URL oui.is/bc2411aa5907. The page displays the results for the MAC address bc:24:11:aa:59:07. The results table has four columns: Organization, MAC Prefix, MAC Range, and Registry. The organization is listed as Proxmox Server Solutions GmbH, the MAC Prefix is bc:24:11:00:00:00/24, and the MAC Range is bc:24:11:00:00:00-bc:24:11:ff:ff:ff. The Registry column shows 'MA-L'. Below the table, there is a search bar with the placeholder '00:53:00:00:b3:3f' and a 'SEARCH AGAIN' button. At the bottom of the page, there is a timestamp '12/8/24, 4:22:30 PM EST' and a note 'Last Updated'.

- Each MAC address is globally unique.
- Device manufacturers are assigned prefixes by the IEEE.
- <https://oui.is> is one site that allows you to look up MAC addresses and find the manufacturer.



Conclusion

- Arp works at Layer 2 of the OSI network model, below most common networking protocols.
- There are several tools which you can use to discover network hosts via the ARP protocol.
- As shown in the previous examples, results can vary depending on the tool, whether you're run it with admin rights, and depending on the switch port to which you're connected. So, don't rely on just one tool or just one network connection.
- Because MAC addresses are unique and ranges are assigned to device manufacturers you can research online who made them, which can help you track down devices on your network.
- Leveraging arp can facilitate you discovering hosts on your network that may escape other detection tools.



Thank you for watching!

Grab a copy of this slide deck from GitHub:

<https://github.com/InformationTechWithDave>

If you liked this video please hit the like button and subscribe.

Subscribe

Don't forget to click the notification button.

 Subscribed ▾

Please share with anyone who would find this helpful.

 Share

@InformationTechWithDave

