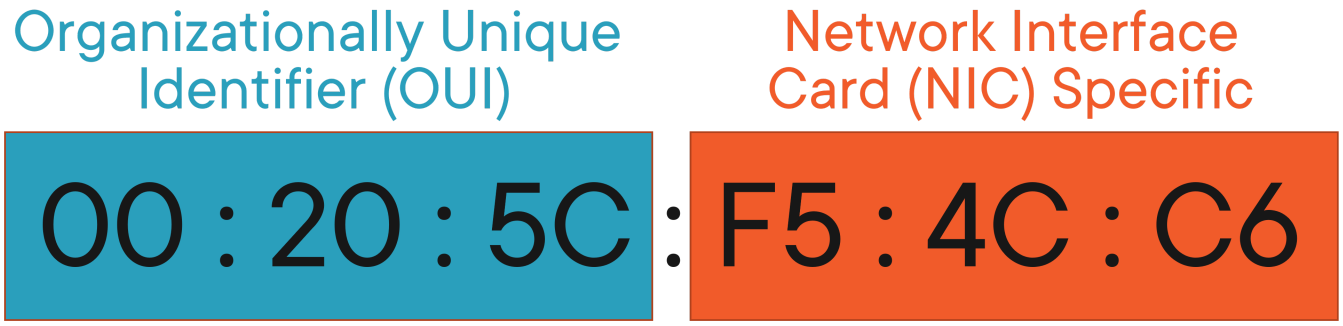
MAC Addressing

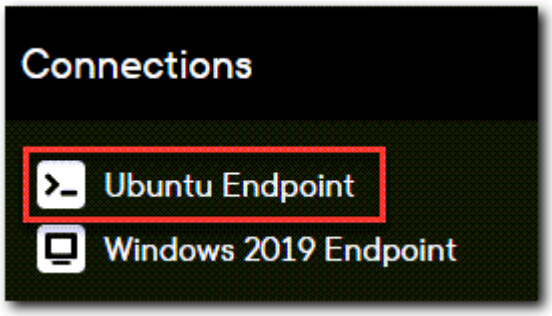
The purpose of this lab is to introduce you to the IPv4 and ARP protocols with an understanding of how the translation between MAC and IP addressing works. This lab focuses on the fundamentals of analyzing the protocol behavior for security professionals.

Starting off, you'll explore Media Access Control (MAC) addresses. This address may also be referred to as the physical address since it's the identifier assigned to a Network Interface Card (NIC). While this is typically a physical card, or controller, that you might plug the ethernet or fiber into, MACs are also used to identify a pseudo-physical address for logical interfaces as well. A good example of this can be seen when looking at virtual machines or docker containers.

The format of a MAC address is six groups of two hexadecimal digits, typically separated by colons:



This first challenge will be using the **Ubuntu Endpoint**:



Important Note: We are building a custom environment just for you! This may take a few minutes depending on its complexity.

- In the terminal environment, typing status will return an overview on applications installing and/or lab files being downloaded. Once this returns **SYSTEM**

COMPLETE, your environment will be ready for full exploration!

1. To begin, use tcpdump to start a packet capture on the docker interface

   ```
   sudo tcpdump -nn -i docker0 -w ~/analysis1.pcap &
   ```

   Note: In order to not interrupt your active connection to the lab environment, you will be modifying the docker interface.

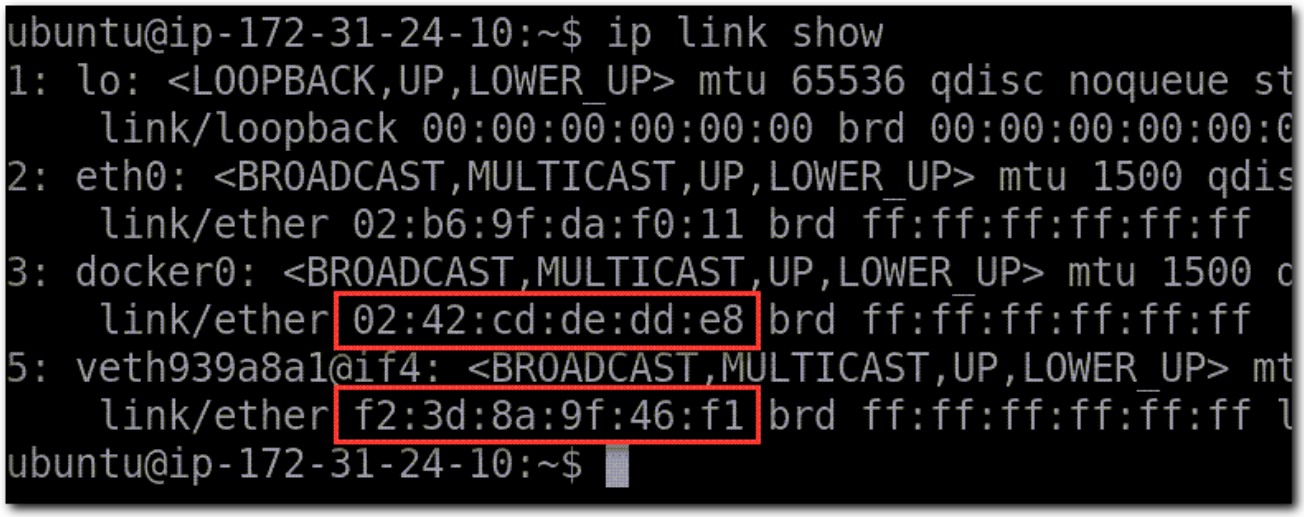
   Note: tcpdump options:

   - `-nn` = No IP or port resolution. For both operational security reasons and to ensure tcpdump isn't presenting you with modified information, these flags are important
   - `-i` = Selects the interface to capture packets from
   - `-w` = Writes the packets to a PCAP file instead of printing to the terminal screen
   - `&` = (Linux command) Runs the command in the background

2. Before you generate any traffic, take a look at the current MAC addresses assigned to your endpoint:

   ```
   ip link show
   ```

   Note: These addresses may be different for your machine.

   Analysis: For this challenge, we will be focusing on the virtual network created between your local endpoint and a web application running locally inside a docker container. The **docker0** interface is your endpoint's interface for communication to docker containers. The "veth..." interfaces are the virtual interfaces for the web applications.

3. You can now generate some sample traffic by sending a request to the web application:

   ```
   curl http://localhost:8000
   ```

   Note: curl is a command-line tool used for making requests between endpoints and applications.

   ```
   ubuntu@ip-172-31-24-10:~$ curl http://localhost:8000
   GET / HTTP/1.1
   Header["User-Agent"] = ["curl/7.58.0"]
   Header["Accept"] = ["*/*"]
   Host = "localhost:8000"
   RemoteAddr = "172.17.0.1:40108"

   ===> local IP: "172.17.0.2"

   Env message = PluralsightRocks!
   ubuntu@ip-172-31-24-10:~$
   ```

4. Even though the MAC address is supposed to be a statically assigned identifier for the specific NIC, they are extremely easy to change. First, you'll need to turn the desired interface off:

   ```
   sudo ip link set dev docker0 down
   ```

   Note: This will cause tcpdump to stop. That's ok! We'll bring it back to life in a minute.

5. Now set the interface to an arbitrary MAC:

   ```
   sudo ip link set dev docker0 address CA:FE:C0:FF:EE:00
   ```

   Note: Typically, attackers will spoof a MAC to mimic a desired type of device or use randomization software to mask their endpoint.

6. Turn the interface back on:

   ```
   sudo ip link set dev docker0 up
   ```

7. Bring tcpdump back to life, this time writing to a new PCAP file:

   ```
   sudo tcpdump -nn -i docker0 -w ~/analysis2.pcap &
   ```

8. Now generate a bit more traffic by performing the same curl command:

   ```
   curl http://localhost:8000
   ```

9. Bring tcpdump back to the foreground with `fg`, then press `CTRL+C` to stop the process

   ```
   ubuntu@ip-172-31-24-10:~$ fg
   sudo tcpdump -nn -i docker0 -w ~/analysis2.pcap
   ^C14 packets captured
   14 packets received by filter
   0 packets dropped by kernel
   ```

10. Before you dive into analysis, merge both PCAP files for easier viewing:

    ```
    mergecap -w combined-analysis.pcap analysis1.pcap analysis2.pcap
    ```

    Note: mergecap is part of the Wireshark toolset. This allows you to combine any number of PCAP files into a single file.

11. Take a broad look at the network traffic by listing out all the packets:

    ```
    tcpdump -nn -r ~/combined-analysis.pcap -e | less -S
    ```

    Note: The `-e` option in tcpdump prints out the MAC addresses

    Note: The `less -S` command in Linux allows you to read longer files by using the arrow keys and disabling word-wrap. Once you are finished analyzing a file, you can press `q` to quit

    ```
    02:42:cd:de:dd:e8 > 02:42:ac:11:00:02, ethertype IPv4 (0x0800), length 74: 172.17.0.1.40112 > 172.
    02:42:ac:11:00:02 > 02:42:cd:de:dd:e8, ethertype IPv4 (0x0800), length 74: 172.17.0.2.8000 > 172.1
    02:42:cd:de:dd:e8 > 02:42:ac:11:00:02, ethertype IPv4 (0x0800), length 66: 172.17.0.1.40112 > 172.
    02:42:cd:de:dd:e8 > 02:42:ac:11:00:02, ethertype IPv4 (0x0800), length 144: 172.17.0.1.40112 > 172
    02:42:ac:11:00:02 > 02:42:cd:de:dd:e8, ethertype IPv4 (0x0800), length 66: 172.17.0.2.8000 > 172.1
    02:42:ac:11:00:02 > 02:42:cd:de:dd:e8, ethertype IPv4 (0x0800), length 383: 172.17.0.2.8000 > 172.
    02:42:cd:de:dd:e8 > 02:42:ac:11:00:02, ethertype IPv4 (0x0800), length 66: 172.17.0.1.40112 > 172.
    ```

    Analysis: You can start to match up MAC addresses with their associated IP addresses. tcpdump output will have the sources listed first and destinations listed second.

12. In security, you are interested in any time MAC addresses arbitrarily change. The above is a very manual method for matching these up, so let's take a look at a cleaner output:

```
tcpdump -nn -r ~/combined-analysis.pcap -e 'tcp port 8000' | cut -f 2,10 -d " " | cut -f 1-4 -d "." | sort | uniq -c | sort -n
```

```
reading from file /home/ubuntu/combined-analysis.pcap,
      6 02:42:cd:de:dd:e8 172.17.0.1
      6 ca:fe:c0:ff:ee:00 172.17.0.1
      8 02:42:ac:11:00:02 172.17.0.2
```

Analysis: From the output, you can see how many times a packet was sent from each unique set of MAC and IP addresses. With typical enterprise endpoints, you shouldn't expect these to change very often. Basically, each endpoint within an environment has a single IP and there should be one MAC associated with that. Even when using DHCP, endpoints will stay consistent unless there is a very fast turnover for devices (Public Wi-Fi, hotels, conference areas, etc.). There are some exceptions, but you'll take a look at that in more detail during the next challenge.

Whoa! Ok, there's a lot happening in that command, so let's break it down:

```
cut -f 2,10 -d " "
```

The cut command allows you to extract out columns (or fields) of text. In this case, it pulls out the source MAC address and source IP address columns.

`-f` = specifies the field (or column) number. Multiple fields can be selected by using a comma or a dash.

`-d` = specifies the delimiter to split the fields by. This example is using a space, encapsulated by quotation marks.

```
cut -f 1-4 -d "."
```

The tcpdump output combines the IP address with port numbers separated by dots (ex. 172.31.24.10.8080). In order to remove the ports, you have to extract out the first 4 columns, using a dot as the delimiter.

```
sort | uniq -c | sort -n
```

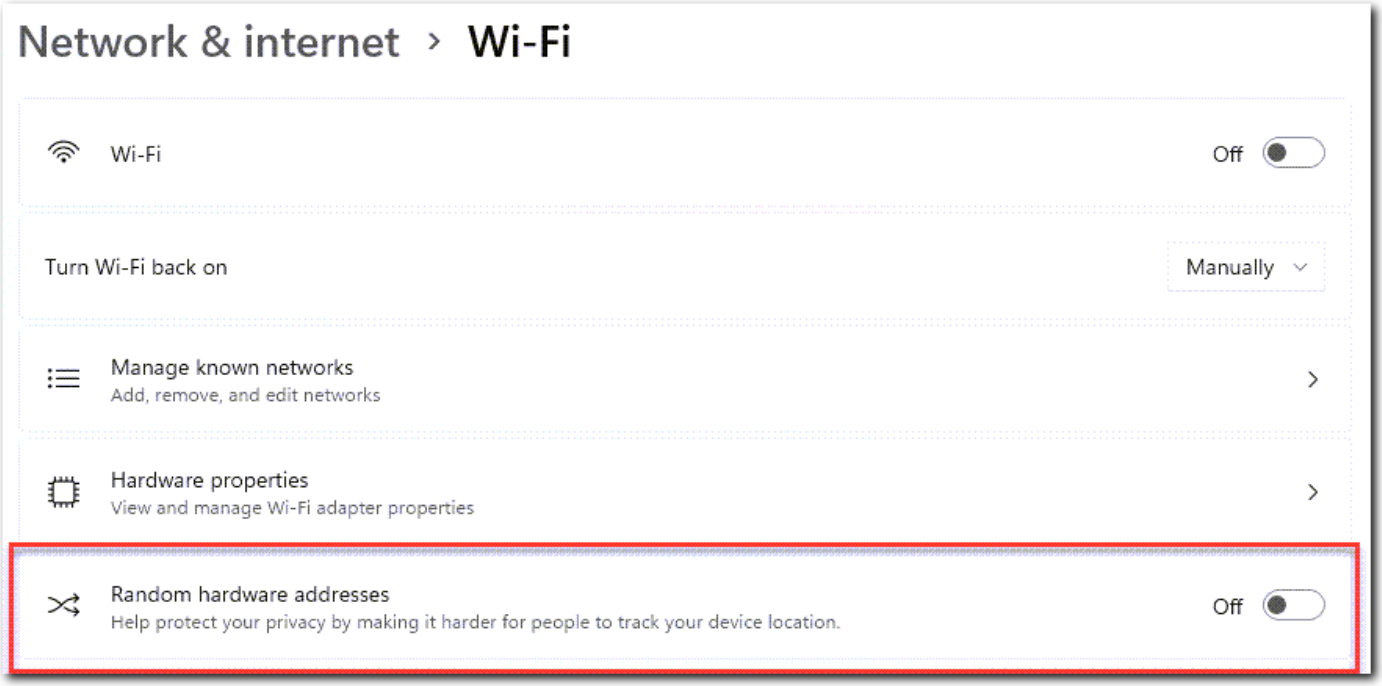
The sort and uniq commands allow you to filter command line output and return a cleanly sorted and unique list of results. You must sort the list first since the uniq command follows a top-down approach and will display unexpected results on an un-sorted list.

`uniq -c` = provides a unique list of values and presents a count in the left-column on how many times that value was seen

`sort -n` = sorts a list numerically with the highest values at the bottom

Reference: https://www.devaultsecurity.com/blog/2021/09/04/analyzing-logs-with-the-linux-sort-sandwich/

As a bonus note, in Windows 10 and 11, it's as simple as clicking a check-box to randomize your MAC address!



In the next challenge, you will expand the analysis on the traffic you just created by taking a look at the protocol used to make the translation between MAC and IP.