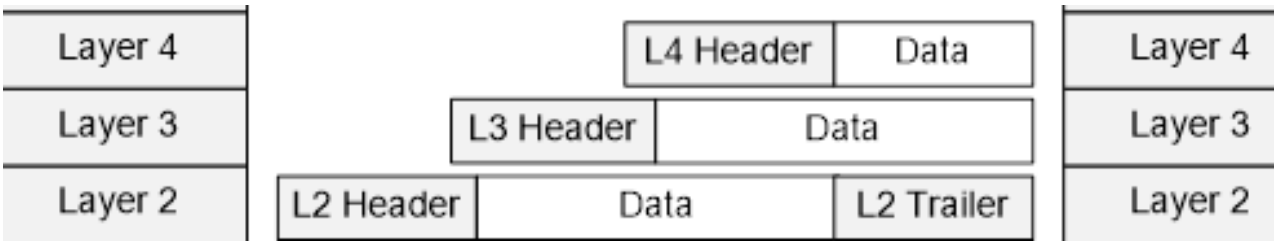
# DAY 5 - Ethernet LAN Switching

Purinat33

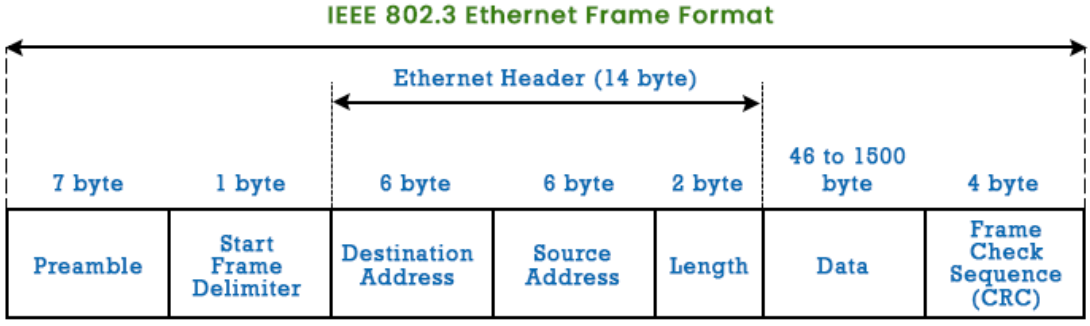
## Ethernet LAN Switching

- **Before**: We have looked at **Physical Layer Ethernet** (UTP, Fiber Optics etc.).
- **Now**: We will look at **Data Link Layer Ethernet (Frame)** now.




---

### Ethernet Frame:



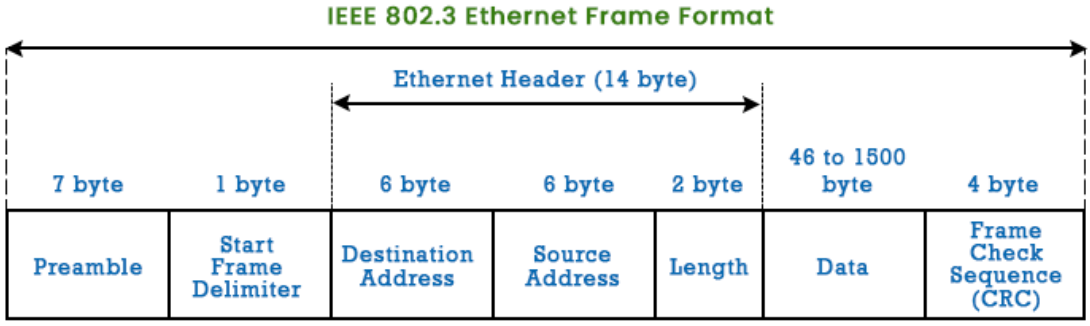

- **Minimum Size**: 64 Bytes (Not including **Preamble + SFD**)
- **Minimum Data Payload Size**: 46 Bytes
  - If the size is less than 46, `0` will be padded to the Data Payload until it is 46 Bytes long.

#### 1. Ethernet Header

- **Preamble**:
  - Not really part of the Ethernet header.

  - 7 bytes of consecutive `10101010` s.
  - Use for synchronization of receiver clocks.
- **SFD** (**Start Frame Delimiter**):
  - Also not really part of the Ethernet header.
  - 1 byte of `10101011`
  - Marks the end of the preamble, and the beginning of the rest of the frame.
- **Destination/Source Address**:
  - Indicate devices sending & receiving the frame.
  - **MAC Address** (Media Access Control):
  - 6 bytes each
- **Type** or **Length**:
  - 2 Bytes field.
  - Depending on the value inside this field:
    * A value of **1500 or less** in this field indicates the **LENGTH** of the *encapsulated packet* (in Bytes)
    * A value of **1536 or greater** indicates the **TYPE** of the *encapsulated packet*. The length of the packet will be determined via other methods.
      · `0x0800` : **IPv4**
      · `0x86DD` : **IPv6**
      · `0x0806` : **ARP**

**2. Ethernet Trailer**




- **FCS**:
  - Frame Check Sequence
  - 4 Bytes
  - Detects corrupted data by running a CRC Algorithm over the received data.

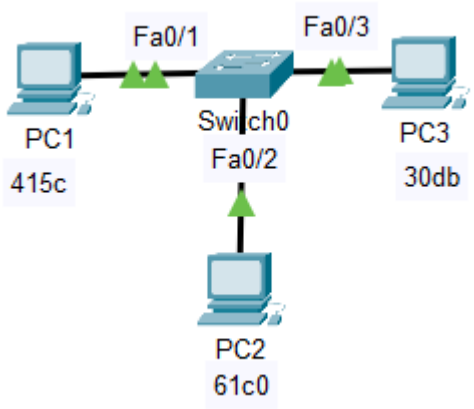
---

## MAC Address

- 6 Bytes (48 bits) physical address assigned to the device when manufactured
- Also known as **Burned-In Address**
- Is globally *unique*
- Written as 12 Hexadecimal digits
  - C8-3A-35-C8-01-04
  - C83A.35C8.0104
- The **First 3 Bytes** are the **OUI** (Organizationally Unique Identifier), which is associated to the company making the device.
- The **Last 3 Bytes** are unique to the device itself.
  - C8-3A-35-C8-01-04
    - `C8-3A-35` : **OUI**
    - `C8-01-04` : **Device**

---

## Frame Transmission




**Switch**: Forwards Frames out of its various interfaces to the frame intended destinations based on the **Switch**'s MAC Address Table.
(eg.):

```
Switch#show mac address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    000c.cf25.415c    DYNAMIC     Fa0/1
   1    0060.2fce.30db    DYNAMIC     Fa0/3
```

**Unicast Frame**: A frame destined for a single target. (eg. PC1-PC2)

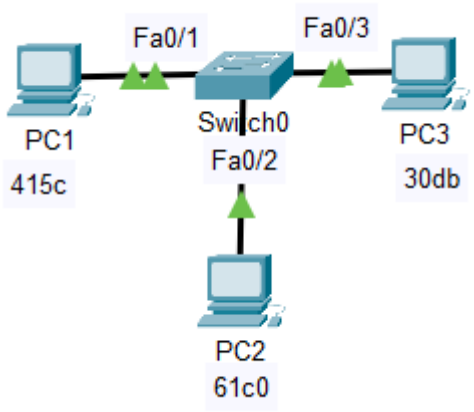
- **Known Unicast Frame**: The **Destination MAC Address** of the Frame is found in a switch's **MAC Address Table**. The switch will simply only forward the frame out of the matching port/interface.
  - **Source:** .415c
  - **Destination**: .30db
  - **Send out Port**: Fa0/3

```
0060.2fce.30db     DYNAMIC     Fa0/3
```

- **Unknown Unicast Frame**: The **Destination MAC Address** of the Frame is not found in the switch's **MAC Address Table**. The switch will need to learn what port is associated with what MAC Addresses.

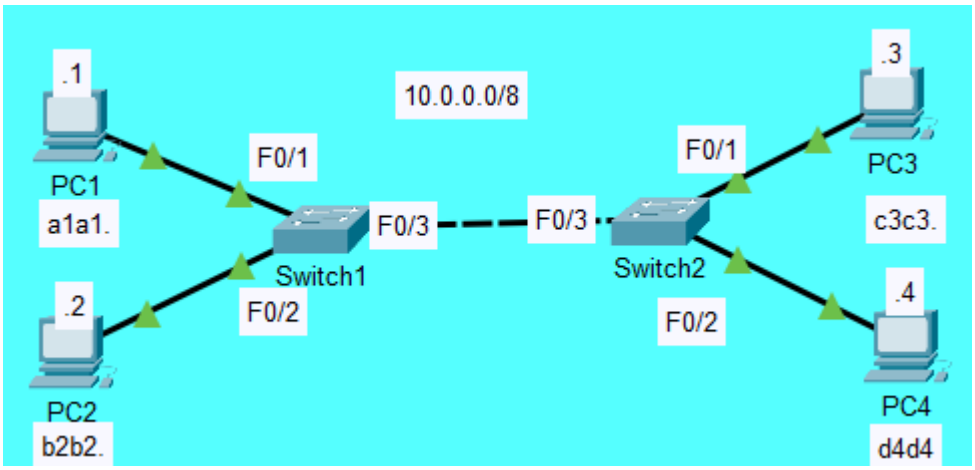
---

## MAC Address Learning (PC and Switch)




- Normally devices communicate using **IP Addresses** and not **MAC Addresses**.
  eg. `ping 192.168.1.1` and not `ping a9-12-00-3c-10-f9`
- How will a Source device learn the **MAC address** of the Destination device?
- Answer: Using Address Resolution Protocol (**ARP**)

### Address Resolution Protocol (ARP)




1. PC1 wants to communicate with PC4.
2. PC1 only knows the **IP Address** of PC4.
3. PC1 cannot send an Ethernet frame yet since the **Destination MAC Address** is unknown.
   (a) **Source IP Address**: `10.0.0.1`
   (b) **Destination IP Address**: `10.0.0.4`
   (c) **Source MAC Address**: `a1a1.` (abbreviated)
   (d) **Destination MAC Address**: `UNKNOWN` (not in database)
4. PC1 use **ARP** to discover PC4's MAC Address.
5. PC1 creates an **ARP REQUEST Frame** which broadcast to every devices in the LAN.
   (a) **Source IP Address**: `10.0.0.1`
   (b) **Destination IP Address**: `10.0.0.4`
   (c) **Source MAC Address**: `a1a1.`
   (d) **Destination MAC Address**: `FFFF.FFFF.FFFF` (broadcast)
6. PC1 basically shouts "Whoever has an IP of `10.0.0.4` please reply directly to `a1a1.` with your MAC Address".
7. The **ARP REQUEST Frame** enters *Switch 1* via the `F0/1` interface.
8. *Switch 1* discovers that:
   (a) **A Frame entered its F0/1 interface**
   (b) **That Frame has a Source MAC Address of** `a1a1.`
   (c) **THUS, associate any frame with MAC Address** `a1a1.` **with interface** `F0/1` **into its MAC table.**
9. *Switch 1* sees that the **Destination MAC Address** is `FFFF.FFFF.FFFF` which is the *broadcast address*.
10. *Switch 1* duplicates and **FLOOD** the frames through all of its ports except the incoming port.
11. *Switch 2* and PC2 receives the frames, PC2 sees that the **Destination IP**

**Address** ( `10.0.0.4` ) wasn't meant for them, and thus discard the frame.

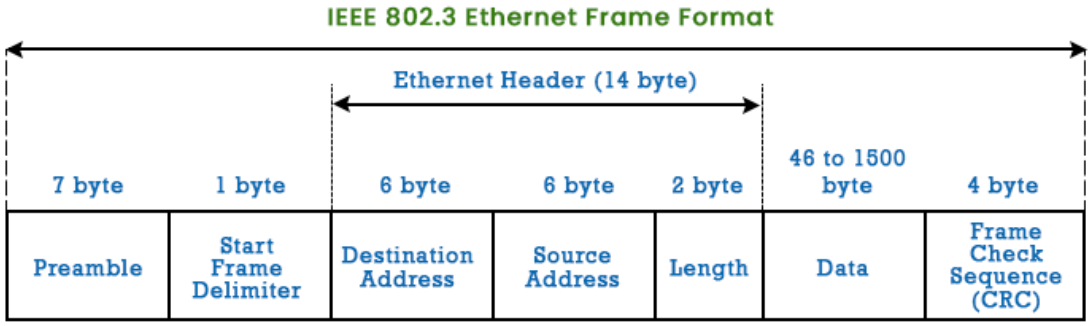
12. *Switch 2* perform the same thing as *Switch 1* including flooding the frames and adding `a1a1.` with `F0/3` into its MAC table.
13. PC3, PC4 all receive the **ARP Request Frame**
14. PC3 sees that the **Destination IP Address** ( `10.0.0.4` ) wasn't meant for them, and thus discard the frame. Only PC4 accepts the frame.
15. PC4 creates an **ARP Reply** which is a **unicast** Frame this time, using the information from PC1 that comes with the **ARP Request**.
    (a) **Source IP Address**: `10.0.0.4`
    (b) **Destination IP Address**: `10.0.0.1`
    (c) **Source MAC Address**: `d4d4.`
    (d) **Destination MAC Address**: `a1a1.`
16. The reverse process is basically the same as the **ARP Request** process.
17. When PC1 receives the **ARP Reply** unicast frame from PC4, PC1 adds PC4's MAC Address to its own memory called an **ARP Table**
    (a) can be viewed using `arp -a` on a device or `show arp` on Cisco devices.

```
C:\>arp -a
  Internet Address      Physical Address      Type
  10.0.0.4              0001.43c5.4caa        dynamic
```

---

# Summary

## Ethernet Frame




**Pre-Header**:

- **Preamble**: 7 Bytes of `10101010` for synchronization.
- **SFD**: `10101011` , marks the end of the *Preamble*

**Ethernet Header (14 Bytes)**:

- **Destination MAC Address**:

  - 6 Bytes
  - If Destination MAC Address is `FFFF.FFFF.FFFF` = Broadcast to all devices in the **LAN** (except to the originator)
- **Source MAC Address**:
  - 6 Bytes.
- **Length/Type**:
  - 2 Bytes
  - If value is **Less than or equal 1500** indicates the **LENGTH** of the data payload.
  - If value is **More than or equal 1536** indicates the **TYPE** of the data payload.
    - `0x0800` : **IPv4**
    - `0x86DD` : **IPv6**
    - `0x0806` : **ARP**

**DATA PAYLOAD**:

- 46 to 1500 Bytes
- If the length is less than 46, then `0` bits are padded to the data payload until the length is 46.

**Ethernet Trailer (4 Bytes)**:

- **Frame Check Sequence (CRC)**: Detects corruption.

## MAC Address

- 12 Hexadecimal Digits
- 6 Bytes (48 bits)
  - First 3 Bytes (6 Hexadecimal Digits): **OUI** (Organizational)
  - Last 3 Bytes (6 Hexadecimal Digits): Unique to a device.

## Switches

- View/Clear MAC table of a switch.
  - `show mac address-table`

```
Switch#show mac address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  20    001b.10a0.2500    DYNAMIC     Gi1/0
  20    001b.10ae.7d00    DYNAMIC     Gi0/0
  30    001b.108c.8700    DYNAMIC     Gi1/2
  30    001b.10ae.7d00    DYNAMIC     Gi0/0
  30    0050.7966.6803    DYNAMIC     Gi1/1
```

  - `clear mac address-table dynamic address {address}` Clear the entry with a specific **MAC Address**.
  - `clear mac address-table dynamic interface {interface}` Clear the entry with a specific **PORTS**
- Switch learns the MAC/Ports mapping via **Source MAC Address** of an **Incoming Frame** to the **Port** it came through.
- **Flooding**: The act in which Switch duplicates and then send out the duplicated Frames out of all the ports **except** the original incoming port.
  - Broadcast
  - Unicast with unknown **Destination MAC Address** not in the table.
- **Aging**: The act in which after some times have passed, the entries which are *old* enough will be removed from the table.

## Address Resolution Protocol (ARP)

- The sender/source usually doesn't know the **MAC** of the receiver/destination device. Only the **IP** is known.
- The sender creates an **ARP Request** Broadcast frame, containing the:
  - Sender's IP Address
  - Receiver's IP Address
  - Sender's MAC Address
  - **Broadcast MAC Address** ( `FFFF.FFFF.FFFF` )
- Every devices in the **same LAN** will receive the **ARP Request** Frame, but only the intended receiver will accept the frame (based on the **Receiver's IP Address**).
- The receiver creates an **ARP Reply** Unicast frame, containing:
  - Receiver's IP Address
  - Sender's IP Address
  - **Receiver's MAC Address**
  - Sender's MAC Address
- A device stores its known **MAC/IP** mapping in its **ARP table**
  - Viewed using `show arp` or `arp -a`

---