# ETHERNET

**Ethernet:** Basic frame format/address schemes are the same for all varieties:

- Operates on DLL/physical: IEEE 802.2/802.3 standards

**Data bandwidth support:**

| | | | | | |
|---|---|---|---|---|---|
| 10Mbps | 100Mbps | 1000Mbps (1Gbps) | 10,000Mbps (10Gbps) | 40,000Mbps (40Gbps) | 100,000Mbps (100Gbps) |

**Relies on 2 sublayers in DLL to operate:**

| | |
|---|---|
| **LLC: Logical Link Controller** | **Communication bet upper/lower layers**<br>• Takes protocol data (IPv4 packet) + adds control info<br>• Helps deliver to destination<br>• Talks w/upper layers/transitions to lower layers for delivery<br>• Implemented in software \|\| Independent of hardware<br>• Driver software for NIC<br>• Driver interacts w/h-w directly to pass data bet. MAC/physical media |
| **MAC: Media Access Control** | **Lower portion of DLL:** Implemented by hardware (NIC)<br>• Specifics noted by IEEE 802.3 standards<br>**2 primary responsibilities:**<br>1. Data encapsulation<br>2. Media Access Control |

**Ethernet MAC 2 Responsibilities:**

| | |
|---|---|
| **Data encapsulation** | Frame assembly before transmission<br>• Frame disassembly on reception of frame<br>• MAC adds header/trailer to network layer PDU<br>• Protocol Data Unit (frames)<br>**Data encapsulation: 3 primary functions:**<br>1. Frame delimiting<br>2. Addressing<br>3. Error correction |
| **Media Access Control** | Places/Removes frames on/from media<br>• Communicates directly with physical layer<br>• Ethernet a multi-access bus/All nodes share medium<br>• Ethernet uses contention-based method & CSMA |

**Data Encapsulation 3 functions:** The use of frames aids transmission of bits

| | |
|---|---|
| **Frame delimiting** | Delimiters to ID groups of bits that make up frame<br>• Sync bet transmitting/receiving nodes |
| **Addressing** | Each header added in frame contains MAC address<br>• Enables frame delivery to destination node |
| **Error detection** | Each frame contains trailer with **CRC: Cyclic Redundancy Check**<br>• After reception, receiving node creates CRC to compare frame<br>• If 2 CRC calcs match: Frame trusted to be received w/out error |

**CSMA:** Ethernet uses CSMA (contention-based)

1. Tries to detect if media carrying signal
2. If signal from another node detected: Another device transmitting
3. Device waits to connect if so
4. No signal? Device transmits
5. If process fails/2 devices transmit at same time: Collide/fail

**Contention-based:** Doesn't require mechanisms to track whose turn it is to access media
**CSMA usually implemented w/ways of solving media contention**

| | |
|---|---|
| **CSMA CD** | **Carrier Sense Multiple Access/Collision Detection**<br>• Device monitors media for data signal<br>• If absent: Transmits data<br>• If 2 present at same time: They both stop<br>• Traditional forms of Ethernet transmission<br>**Today: Almost all wired connections in Ethernet LAN are full-duplex**<br>• Devices send/receive simultaneously<br>• Ethernet is designed with CSMA/CD, but not necessary<br>• Intermediate devices don't have collisions |
| **CSMA CA** | **Carrier sense multiple access/Collision Avoidance**<br>• Wireless connections in LANs take collisions & use CSMA/CA<br>• Devices examine media for presence of data signals<br>• If media free: Device sends notification across media with intent of use<br>• The device then sends data<br>Used by 802.11 wireless |

**MAC Address: Ethernet Identity**: Ethernet is a logical topology with multi-access bus
- Every device connected to same shared media: ALL nodes receive ALL frames transmitted
- To prevent too much overheard in processing every frame: MAC addresses created

**MAC addressing:** Provides identify to actual source/destination nodes within an Ethernet network
- Provided method for device ID at lower OSI lvl
- 48 bit bin values expressed as 12 hex digits (4 bits per digit)
- Added a part of a layer 2 PDU

**MAC address structure:** Must be globally unique: IEEE rules ensure so: Req vendors that sell Ethernet devices to register **Organizationally Unique Identifier:** IEEE Assigns vendors a 3byte (24bit) code (OUI)
**IEEE requires vendors to follow 2 rules:**
1. 1st 3 bytes: MAC addresses assigned to NIC/Ethernet devices use vendor's assigned OUI
2. Last 3 bytes: MAC addresses with same OUI must be assigned a unique value (code/serial)

**BIA (burned-in addresses):** MAC addresses referred as BIA's b/c they were burned into ROM on NIC
- Address encoded into ROM chip permanently: Couldn't be changed by software

**On modern OS's/NICs, you can change MAC addresses in software:**
- Helpful when trying to gain access to a network that filters based on BIA
- Filtering/controlling traffic based on MAC addresses isn't secure

**Frame Processing:** MAC addresses assigned to devices that must start/receive data on Ethernet network
- Workstations/servers/printers/switches/routers
- Software/hardware manufacturers use addresses in different hex formats

| | | | |
|---|---|---|---|
| **Examples** | 00-55-9A-3C-78-00 | 00:05:9A:3C:78:00 | 0005.9A3C.7800 |

| | |
|---|---|
| **When PC starts** | NIC copies MAC address from ROM into RAM<br>**When a device forwards a message (Ethernet):**<br>• It attaches header info to packet<br>• Header info has source/dest MAC<br>• Source device sends data/each NIC views at MAC sublayer<br>• Frames checked to match physical address stored in RAM<br>• No match? Device discards frame<br>**When frames reach dest**: NIC passes frame up OSI where de-encapsulation happens |

**Ethernet Versions:**

| Year | Standard | Description |
|---|---|---|
| 1973 | Ethernet | Dr. Robert Metcalf of Xerox Corp. invents Ethernet |
| 1980 | DIX Standard/Ethernet II | Digital Equipment Corp./Xerox/Intel release a standard for 10-Mbps Ethernet over coax |
| 1983 | IEEE 802.3/10Base5 | 10Mbps Ethernet over thick coax |
| 1985 | IEEE 802.3a/10Base2 | 10Mbps Ethernet over thin coax |
| 1990 | IEEE 802.3i/10Base-T | 10Mbps Ethernet over TP (twisted pair) |

| | | |
|---|---|---|
| 1993 | IEEE 802.3j/10Base-T | 10Mbps Ethernet over fiber-optic |
| 1995 | IEEE 802.3u/100Base-xx | Fast Ethernet: 100Mbps over twisted paid/fiber |
| 1998 | IEEE 802.3z/1000Base-X | Gigabit Ethernet over fiber-optic |
| 1999 | IEEE 802.3ab/1000Baste-T | Gigabit Ethernet over twisted pair |
| 2002 | IEEE 802.3ae/10G Base-xx | 10 Gigabit Ethernet over fiber |
| 2006 | IEEE 802.3an/10G Base-T | 10 Gigabit Ethernet over twisted pair |

**Ethernet Encapsulation:** At DLL, frame structure is almost identical for all speeds of Ethernet

- Frame structure adds headers/trailers around L3 PDU to encapsulate msg
- Ethernet header/trailer have 7 sections of info used by protocol

**Frame:** Each section of information used by the Ethernet protocol

**2 styles of Ethernet framing:**

1. **IEEE 802.3 Ethernet standard (updated several times to include new tech)**
2. **DIX Ethernet standard: Referred to as Ethernet II**

**Differences:**

1. Addition of SFD (start frame delimiter)
2. Type field changed to length field in 802.3

**Ethernet Frame Size:** Ethernet II/IEEE 802.3: Min frame size: 64 bytes Max frame size: 1522 bytes

**Collision fragment or runt frame:** Any frame less than 64 bytes length (auto discarded by receivers)

**VLAN:** 1998: Extended max allowable frame size to 1522 bytes (created in switched network)

**QoS:** Additional fields: AKA Tags: Specified by IEEE 802.1Q, inserted into Ethernet frame

- Enable VLAN/other info to be included
- If size of transmitted frame less/greater than min/max receiving device drops frame

**At the DLL:** Frame structure is nearly identical

**At physical layer:** Different versions of Ethernet vary in methods of detecting/placing data on media

**Ethernet Frame Fields:**

| | |
|---|---|
| **Preamble/SFD: Start Frame Delimiter** | **Preamble: 7 bytes: SFD: 1 byte:** Sync bet sending/receiving devices<br>• 1st 8 bytes of frame: Gets attn of receiving nodes<br>• Tell receivers to get ready to receive new frame |
| **Destination MAC** | **6 byte field:** ID for recipient: L2 determines<br>• Compared to MAC of devices: If match: Accepts frame |
| **Source MAC** | **6 byte field:** ID's frame's originating NIC/int |
| **Length** | **Any IEE 802.3 standard before 1997:**<br>• Defines exact length of frame's Data field<br>• Used to ensure msg was received right<br>• Which higher-layer protocol present<br>**If 2 octet value =/greater than 0x0600 hex/1536 decimal**<br>• Contents decoded according to Ethernet protocol<br>**If 2 octet value =/less than 0x05DC hex/1500 decimal**<br>• Field used to indicate use of IEEE 802.3 format<br>***How Ethernet II/802.3 frames differentiate*** |
| **Data** | **46 to 1500 bytes:** Encapsulate data from higher layer (L3 PDU/IPv4 packet)<br>• All frames must be at least 64 bytes long<br>• If packet encapsulated: Addl bits called pad used<br>• Pads increase size of frame to min size |
| **FCS: Frame Check Sequence** | **4 bytes:** Detects errors in frame w/CRC<br>• Sending device receives frame/generates CRC for errors<br>• If calcs match: No error<br>• If calcs don't match/data changed: Frame dropped |

**MAC Addresses/Hex**

**Hexadecimal:** Hex number system: Base 10 binary system and Base 16 number system

**Ipconfig /all:** Can be used to identify the MAC address of an Ethernet adapter

- Different MAC addresses are used for layer 2 unicast/broadcast/multicast communications

| | |
|---|---|
| **Unicast MAC** | Unique address: Used when frame sent from single transmitting/dest device<br>• Destination IP must be in IP packet header<br>• Destination MAC must be present in Ethernet frame header<br>• IP/MAC combine to deliver data to 1 specific destination |
| **Broadcast MAC** | Packet contains dest IP: Has all 1's in host portion<br>• Numbering means all hosts on local network receive/process packet<br>• Protocols like DHCP/ARP use broadcasts<br>• Needs corresponding broadcast MAC in Ethernet frame<br>• Ethernet networks: Broadcast MAC 48 1's displayed as hexadecimal<br>**FF-FF-FF-FF-FF-FF** |
| **Multicast MAC** | Allows source device to send packet to group of devices<br>• Devices belonging to multicast group assigned multicast group IP's<br>• Range of IPv4 multicasts is **224.0.0.0 to 239.255.255.255**<br>• Can only be used as destination of packe<br>• Source will always have unicast address<br>Example: Remote gaming/Videoconferencing<br>• Special value begins with 01-00-5E in hex<br>• Remaining address converts lower 23 bits of IP multicast group into 6 hex chars<br>• **01-00-5E-00-00-C8** |

**Two primary addresses assigned to host**

1. Physical: MAC
2. Logical: IP

**MAC on host doesn't change:** Physically assigned: Remains same

**IP similar to address:** Based on where host located: Possible to determine location of where frame should be sent

- IP: Network layer address known as logical b/c nonpermanent/config w/in device software

**End-to-End Connectivity, MAC/IP:**

- Device determines destination through DNS in which IP associated with domain
- Addressing determines end-to-end behavior of IP packet
- Frames accepted/processed on MAC addresses

**ARP: Address Resolution Protocol:** Provides 2 functions:

1. Resolving IPv4 addresses to MAC addresses
2. Maintaining a table of maps

**Resolving IPv4 to MAC addresses:** For frames to be placed on media, they need destination MAC

**ARP table/cache:**

- When packet sent to DLL to be encapsulated into frame:
- Node refers to table in mem to find DLL address mapped to destination IPv4 address
- ARP table stored in RAM

**Each row of an ARP table binds IP to a MAC: We call this relationship a map**

- ARP tables cache mapping for devices on local LANs

**To begin:** Transmitting node attempts to locate MAC mapped to IPv4 destination

- If map found in table: Node uses MAC as destination in frame that encapsulates packet
- Frame then encoded onto networking media

**Maintaining ARP Tables:** ARP tables maintained dynamically with time stamped entries

- If device doesn't receive frame in time: Removed from table

**Two ways a device can gather MAC addresses**

| | |
|---|---|
| **Monitor traffic on local network segment** | • A node receives frames from media<br>• It records source IP/MAC address as a mapping in ARP table<br>• As frames transmit, the device populates the ARP table with address pairs |
| **Sending ARP requests** | • They are layer 2 broadcasts to all devices on Ethernet LAN<br>• They have IP of destination host/broadcast MAC address FFFF.FFFFF.FFFF<br>• Since it's a broadcast/all nodes on Eth0 LAN will receive/look at contents<br>• The node with IP address that matches IP address in ARP request will reply<br>• The reply will be a unicast frame that includes the MAC address |

| | |
|---|---|
| | • It will correspond to the IP address in the request<br>• This response is used to make a new entry in ARP table of sending node |

**ARP uses two different packet types:**

| | |
|---|---|
| **ARP Request** | ARP receives request to map IPv4 to MAC address |
| **ARP Reply** | • Looks for cached map in ARP table<br>• If entry not found: Encapsulation of IPv4 packet fails<br>• Layer 2 processes notify ARP it needs a map<br>• ARP processes send out ARP request packet to discover MAC address of destination device<br>• If device receiving the request has a destination IP, it responds with an ARP reply<br>• A map is created in the ARP table<br>• Packets from that IPv4 address can now be encapsulated in frames<br>• If no device responds to ARP request: Packet is dropped (a frame can't be created)<br>• Encapsulation failure is reported to upper layers of device<br>• If device is intermediary device (router):<br>• Upper layers might respond to source host with an error in ICMPv4 packet |

**Removing ARP Entries:**

- ARP cache timers remove entries that haven't been used for a specific time (Windows stores them for 2 min)
- Times differ depending on device/OS
- If the entry is used again during that time, ARP timer is extended to 10 minutes

**Commands don't invoke execution of ARP:** They just remove entries on table

- ARP service is integrated within the IPv4 protocol and implemented by device

| | | |
|---|---|---|
| **Cisco router** | **show ip arp** | Used to display ARP tables |
| **Windows 7** | **arp -a** | Used to display ARP tables |

**Problems with ARP:** Overhead

**Security:**

**ARP spoofing/poisoning:** Injects wrong MAC address association by issuing fake ARP requests

- Attackers forge MAC addresses, so frames are sent to the wrong destination
- Manually configuring static ARP associations are 1 way to help prevent ARP spoofing
- MAC addresses can be configured on some devices to restrict network access to only listed devices

**Mitigating ARP Problems:**

- Broadcast/security issues related to ARP can be mitigated with modern switches
- Cisco switches support 7 technologies specifically designed to mitigate Ethernet issues w/broadcasts/ARP

**Switches provide segmentation of a LAN:** Dividing the LAN into independent collision domains

- Each port represents a separate collision domain & provides full media bandwidth to node(s) connected on that port
- They don't prevent broadcasts from propagating to connected devices
- They do isolate unicast Ethernet communications so only they are "*heard*" by source/destination devices

**LAN Switches**

- Used in Ethernet networks to improve security/efficiency
- Most LAN switches operated at layer 2, but now layer 3 switches are used
- Makes use of either physical/logical address in a PDU to control flow of information between segments

**Switch Port Fundamentals:**

| | |
|---|---|
| **LOGICAL topology** | Ethernet network is a multi-access bus<br>• All devices share access on the same medium<br>• Logical topology determines how hosts on a network view/process frames sent/received |
| **PHYSICAL topology** | Most Ethernet networks use star/extended star<br>• The Ethernet networks end devices are connected in a point-to-point basis to a layer 2 LAN switch |

**Layer 2 LAN switch:**

- Performs switching/filtering based on layer 2 MAC address

- Transparent to network protocols/user applications
- Builds a MAC address table/uses it to make forwarding decisions
- Depend on routers to pass data between independent IP subnetworks

**Switch MAC Address Table:**
**Switch fabric:** Integrated circuits/machine code that allows data paths through the switch to be controlled

- To know which port to use to transmit a unicast frame, it must learn which nodes exist on each of its ports
- Switches build MAC address tables by recording MAC addresses of nodes connected to each port
- After a MAC address is recorded in the table, the switch knows to send traffic for that node

**MAC Addressing and Switch MAC Tables:**

1. Switch receives a broadcast frame from PC1/Port 1
2. Switch enters source MAC address/switch port that received the frame into address table
3. The destination is a broadcast, the switch floods the frame to all ports (except receiving port)
4. Destination device replies to broadcast with a unicast frame addressed to PC1
5. Switch enters source MAC address of PC2/port of switch that received frame into address table.
    - The destination address of the frame/port are found in the MAC address table
6. Switch can now forward frames between source/destination without flooding

**CAM table: Content addressable memory:** Another term for MAC address table
**Duplex Settings:**

- Switches can operate in different modes
- Ports on a switch must be configured to match duplex settings of the media type

**Two types of duplex settings used for communications on an Ethernet network**

1. Half duplex
1. 2. Full duplex

**Half duplex:**

- Relies on unidirectional data flow (sending/receiving are not done simultaneously)
- Half-duplex implements CSMA/CD to reduce collisions/detect when they happen
- Performance issues: Waiting/Hanging
- Typically older hardware like hubs

**Full duplex:**

- Data flow is bidirectional (simultaneous) so it can be sent/received at the same time
- Enhances performance by reducing wait time
- Ethernet/Fast Ethernet/Gigabit Ethernet NICs offer full-duplex capability
- Collision circuit is disabled b/c end nodes use 2 separate circuits in the cable
- Each full-duplex connection only uses 1 port

**Cisco Catalyst switch supports 3 duplex settings:**

1. Full option sets full-duplex
2. Half option sets half-duplex
3. Auto option sets auto negotiation of duplex

- With this enabled, 2 ports communicate to decide the best mode of operation

Fast Ethernet/10/100/1000 ports: Default is auto
**100BASE-FX**: Default is full
**10/100/1000:** Ports operate in either half/full when set to 10/100Mbps. When set to 1000Mbps: Only operate in full
**Auto-MDIX**
**–** It's necessary to have the correct duplex setting and cable type defined for each port
– Connections between specific devices (switch/switch or switch/router), once required a crossover/straight-through
– Most switches now support the **mdix auto** interface configuration command in CLI
– This enables the automatic medium-dependent interface crossover (MDIX) feature
**When auto-MDIX is enabled:** Switch detects required cable type for copper Ethernet/configures interfaces
**Auto-MDIX is enabled by default on:** Switches running Cisco IOS 12.2(18)SE or later
**Auto-MDIX is disabled by default between:** Cisco IOS 12.2(14)EA1 and 12.2(18)SE
**Frame-Forwarding Methods on Cisco Switches:**
Switches used 1 of the following forwarding methods for switching data between network ports:

1. Store-and-forward switching
2. Cut-through switching

| | |
|---|---|
| **Store-and-Forward Switching** | **When the switch receives frame:**<br>• Stores data in buffers until complete frame is received |

| | |
|---|---|
| | **During the storage process:**<br>• Switch analyzes frame for information about destination- It performs an error check using the CRC trailer portion of the Ethernet frame<br>• If an error is detected: It discards the frame<br>• Reduces amount of bandwidth consumed by corrupt data<br>• Required for QoS on converged networks where prioritization is necessary<br>**Example:** VoIP data streams need priority over web browsing traffic<br>• Sole forwarding method used on current models of Cisco Catalyst switches |
| **Cut-Through Switching** | • Switch acts upon data as soon as received, even if incomplete<br>• Buffers just enough of a frame to read destination MAC<br>• To determine which port to forward the data on<br>• Destination MAC is located in the first 6 bytes of the frame following the preamble<br>• Switch looks up destination MAC in switching table, determines outgoing interface port<br>• Forwards frame onto destination through designated switch port<br>• No error checking on the frame<br>• Faster than store-and-forward, but forwards corrupt frames through network<br>• Corrupt frames consume bandwidth<br>• Destination NIC eventually discards corrupt frames |

**Two variants of cut-through switching:**

| | |
|---|---|
| **Fast-forward:** | • Lowest level of latency<br>• Immediately forwards a packet after reading destination address<br>• Might be times when packets are relayed with errors<br>• The destination network adapter discards faulty packets on receipt<br>• Latency is measured from first bit received to first bit transmitted<br>• Typical cut-through method of switching |
| **Fragment-Free:** | • Switch stores first 64 bytes of frame before forwarding<br>• Compromise between store-and-forward/fast-forward switching<br>• Stores first 64 bytes because most network errors/collisions occur there<br>• Tries to enhance fast-forward by performing a small error check on those 64 bytes<br>**Compromise between:**<br>• High latency/integrity of store-and-forward<br>• Low latency/reduced integrity of fast-forward switching |

**Memory Buffering on Switches:** Switches analyzes some or all of packets

- An Ethernet switch can use a buffering technique to store frames before forwarding them
- Can also be used when the destination port is busy (congestion)

**Two methods of memory buffering:**

1. Port Based
2. Shared Memory

| | |
|---|---|
| **Port-Based:** | • Frames are stored in queues linked to specific incoming/outgoing ports<br>• Frame is transmitted only when queue is finished<br>• Possible for a frame to delay transmission of other frames in memory b/c of busy destination/port |
| **Shared Memory:** | • Deposits all frames into common memory buffer that all ports on the switch share<br>• Amount of buffer memory required by a port is dynamically allocated<br>• Frames in the buffer are linked dynamically to destination port<br>• Packet can be received on one port/transmitted on another w/out moving to a different queue<br>• Switch keeps a map of frame-to-port links showing where packets need to be transmitted<br>• Number of frames stored in buffer restricted by size of memory buffer<br>• Permits larger frames to be transmitted with fewer drops<br>• Important for asymmetric switching |

**Asymmetric switching:** Allows different data rates on different ports
**PoE (Power over Ethernet):** Allows switch to deliver power to a device (IP phone/WAPs) over existing Ethernet cabling

**Switch form factors:**

- Forwarding rate defines processing abilities of a switch by rating how much data switch can process per second
- Entry-Layer switches have lower forwarding rates than enterprise-layer switches
- Thickness of switch/port density (number of ports available on a single switch) etc..

**Fixed-Configuration Switches:** You can't add features/options to the switch beyond original contents of it

**Modular Switches:** Come with different-sized chassis/allow installation of line cards/more configurable

**Cisco Module Options for Switch Slots:**

**SFP: Switch Form-Factor Pluggable:** Supports a number of SFP transceiver modules

**Layer 3 Switching & Cisco Express Forwarding:**

**CEF: Cisco Express Forwarding:** Forwarding method decouples strict interdependence between layer 2/3 decisions

**Two main components of CEF operation are:**

1. FIB: Forwarding Information Base
2. Adjacency tables

- FIB: Similar to a routing table.
    - Routers use tables to determine the best path/destination based on the network portion of the destination IP.
    - With CEF: Information previously stored in route cache, is stored in several data structures for CEF switching.
- Data structures provide optimized lookup for efficient packet forwarding
- A device uses FIB lookup table to make destination-based switching decisions w/out accessing route cache
    - Adjacency tables can be built separately from FIB tables
        - This allows both to be built w/out any packets being process switched
        - MAC header rewrites used to forward a packet aren't stored in cache entries
        - Changes in a MAC header rewrite string don't require invalidation of cache entries

**Major types of layer 3 interfaces:**

**SVI: Switch Virtual Interface:** Logical interface on a switch associated with a VLAN

**Routed port:** Physical port on a layer 3 switch configured to act as a router port

**Layer 3 EtherChannel:** Logical interface on a Cisco device associated with a bundle of routed ports

- SVI's/L3 EtherChannels/other logical interfaces on Cisco devices include loopback/tunnel interfaces

| | |
|---|---|
| **no switchport** | Puts the interface into layer 3 mode |