**Local Area Network Addressing + ARP resolution protocol**

**Local Area Networks and MAC addresses**

**MAC** (or LAN / PHYSICAL / ETHERNET)

* A 48-bit addressed burned in a Network Interface Controller (NIC) memory (hex-based).
* Used “locally” to get a frame from one interface to another physically-connected interface, which is in the same network (IP-addressing sense).
* Are administered by the IEEE, of which manufacturers buy a portion of MAC address space.

**MAC address vs. IP address**

MAC address IP address

|  |  |
| --- | --- |
| **Hard-coded** in read-only memory when adaptor is built. | **Configured**, or learned dynamically. |
| Like a **social security number**. | Like a **postal address**. |
| **Flat name space** of 48 bits (e.g. 00-0E-9B-6E-49-76 hex) | **Hierarchical name space** of 32 bits (e.g. 12.17866.9 dec) |
| **Portable**, can move LAN cards from one LAN to another. | **NOT Portable**, depends on IP subnet of where host is attached |
| Used to get packet between interfaces on the same network | Used to get a packet to a destination IP subnet. |

**ARP: Address Resolution Protocol**

**Question:** how to determine an interface’s MAC address, know its IP address?

**ARP table**: each IP node (host, router) on LAN has a table of

* **<IP Address, MAC address, TTL>** where TTL is the time after which address mapping will be forgotten (typically 20mins)

ARP used in the same Local Area Network: Send datagram from A to B

1. A wants to send datagram to B 🡪 B’s MAC address is not in A’s ARP table.
2. A **broadcasts ARP query** packet, containing B’s IP address 🡪 All nodes on LAN receive broadcast packet.
3. B **receives A’s broadcast packet**, replies to A with B’s MAC address 🡪 Frame sent to A’s MAC address.
4. A **caches/saves IP-to-MAC address pair** in its ARP table until TTL is reached 🡪 **soft state**: info timeout unless refreshed

ARP used in different Local Area Networks: Send datagram from A to B via. R

* For A to send a datagram to B, A must know several things:
  + **B’s IP address** – subnet mask discovered via. DHCP
  + **1st hop router R’s IP address** – default router discovered via. DHCP
  + **R’s MAC address** – ARP

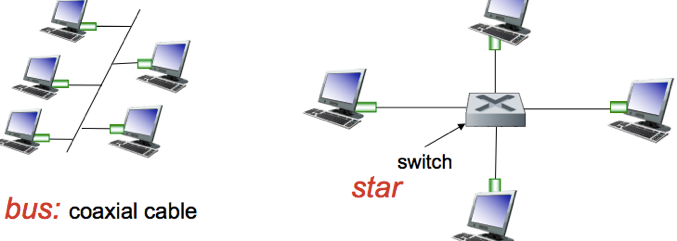
1. A creates IP datagram with src=(A) | dest=(B), and then link-layer frame with dest=(R’s MAC + IP datagram)
2. Link-layer frame is sent from A to R 🡪 Frame is received at R, datagram is detached and passed to IP.
3. R forwards datagram with IP\_src=(A) | IP\_dest=(B) using a forwarding table.
4. R creates a link-layer frame with dest=(B’s MAC + A\_to\_B IP datagram)

**Local Area Networks: Ethernet**

There are many different Ethernet standards, for diff physical layer media w/ diff speeds: fiber, cable etc.

However, they use a common MAC protocol and frame format.

**Ethernet topology**



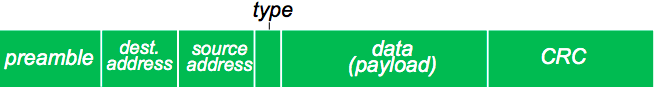
**BUS:** Popular through mid 90s, where all nodes can collide with each other. CSMA/CD for media access control.

**STAR**: Used today, where there is an active SWITCH in the centre.

* Each “spoke” runs a separate Ethernet protocol, so nodes do not collide with each other. No sharing, no CSMA/CD.

**Ethernet Frame Structure**

Sending adaptor encapsulates IP datagram (or other network layer protocol packet) in an **Ethernet Frame**.



**Preamble:** Used to sync receiver and sender clock rates. 7 bytes.

**Addresses**: 6-byte source address + destination MAC addresses.

* If adapter receives frame with matching dest address, or with a broadcast addr (e.g. ARP packet), it passes data in the frame to the network layer protocol.

**Type**: Indicates a higher layer protocol (mostly IP but others possible)

**CRC**: Cyclic redundancy check at the receiver. If error is detected, frame is dropped.

**Ethernet: Unreliable, Connectionless**

**Connectionless**: no handshaking between sending and receiving NICs

**Unreliable**: receiving NIC doesn’t send ACKs/NACKs to the sender NIC. Data in dropped frames recovered only if initial sender uses a higher-layer Reliable Data Transfer (e.g. TCP) otherwise the dropped data is lost.

**Ethernet’s MAC protocol**: un-slotted CSMA/CD with binary backoff.

**Local Area Network: Ethernet Switches**

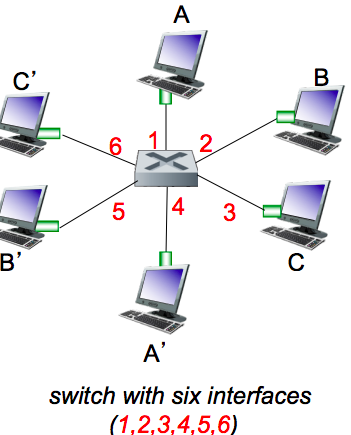
**Link Layer Device**: stores and forwards Ethernet frames.

* Examines incoming frame’s MAC address 🡪 selectively forwards the frame to one or more outgoing links.  
  🡪 uses CSMA/CD to access the segment.

**Transparent**: Hosts are unaware of the presence of switches.

**Plug-and-Play, Self-Learning**: Switches do not need to be configured.

**Switch: multiple simultaneous transmissions + forwarding table**



* Hosts have a dedicated direct connection to the switch.
* Switches buffer packets.
* **Ethernet Protocol** used on each incoming link, but no collisions  
  + full duplex (data can be transmitted in both directions at the same time)

Each link is its own collision domain.

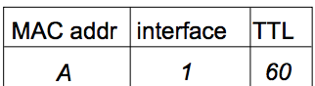
* **Switching**: A-to-A’ and B-B’ can transmit simultaneously without collisions.

Q: How does switch know A’ is reachable via. interface 4, B’ reachable via. interface 5?

A: Each switch has a **Switch Table**, where each entry =  
 **<MAC addr of host, interface to host, time stamp>**

**Switch: Self-Learning + Frame filtering/forwarding**

A switch **learns** which hosts can be reached through which interfaces.

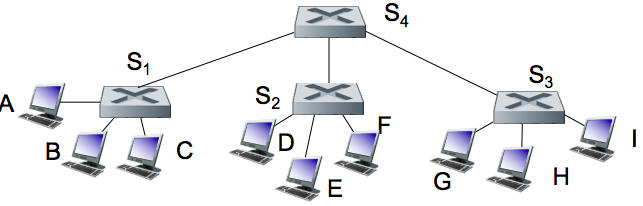
* When frame is received, the switch “learns” location of the sender in the incoming LAN segment.
* The switch records sender/location pair in a switch table.

When a frame is received at the switch:

1. Record the incoming link, MAC address of the sender.
2. Index the switch table using the MAC destination address.
3. **IF** entry is found **{**   
    **IF** dest MAC on segment exists in the switch’s table / comes from same port 🡪 drop/filter packet  
    **ELSE** forward frame on the interface indicated by the entry. **} ELSE** flood /\* forward frame to all the interfaces except the arriving interface \*/

**Interconnecting Switches**

Switches can be connected together.



**Q:** Sending from A-to-G, how does S1 know to forward frame to G via. S4 and S3?  
**A: Self-learning again**! Works exactly the same way as in the single-switch case.

**Switches vs. Routers**

**Switches Routers**

|  |  |
| --- | --- |
| Both are store-and-forward + have forwarding tables. | Both are store-and-forward + have forwarding tables. |
| Link-Layer devices (examine link-layer headers) | Network-Layer devices (examine network-layer headers) |
| Learns forwarding table using **flooding, learning, MAC addresses** | Computes tables using **routing algos, IP addresses** |