Chapter 1

L17 - The Link Layer

Having studied how the network layer routes packets across the Internet, we now examine how individual network segments deliver those packets locally. This brings us to the link layer.

1.1 Fundamentals

1.1.1 Packet Switch Types

Networks contain two fundamentally different types of packet switches:

- **Link-layer switches**: Operate at physical and link layers, forwarding packets within a single network segment using physical addresses
- **Network-layer switches (routers)**: Operate at physical, link, and network layers, routing packets between different networks using IP addresses

Terminology: Throughout this course, "switch" refers to link-layer switches, while "router" refers to network-layer switches.

1.1.2 Scope Comparison: Link vs Network Layer

The key distinction lies in their operational scope:

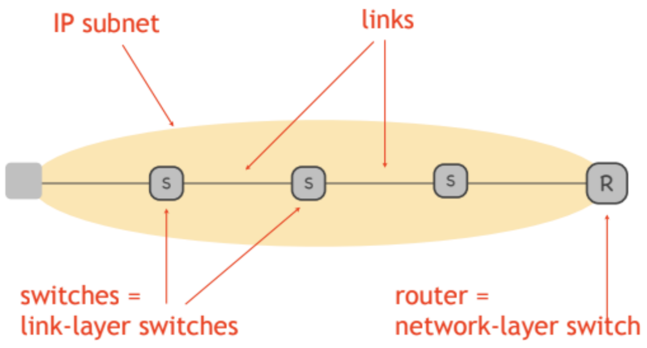
Network Layer: Delivers packets *end-to-end across the entire network* (e.g., New York to Tokyo across the Internet)

Link Layer: Delivers packets *across a single physical link* (e.g., laptop to wireless access point)

Think of the network layer as the postal service routing mail globally, while the link layer is the local delivery truck carrying mail from the post office to your house.

1.1.3 Layer Roles Within IP Subnets

Consider a single IP subnet—essentially one network segment like your home WiFi:

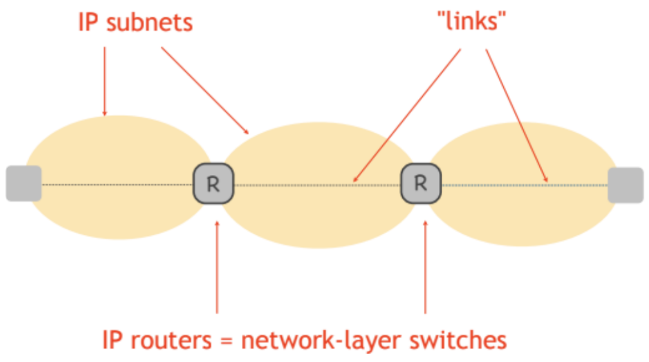


Within this subnet containing end-systems, boundary routers, and interconnecting switches:

- **Link layer:** Moves packets across individual physical connections (one "hop")
- **Network layer:** Moves packets across the entire subnet (multiple hops, e.g., from left computer to router R)

1.1.4 Internet-Scale Architecture

Zooming out to the full Internet reveals multiple interconnected IP subnets:



This layered approach enables Internet scalability:

- Network layer handles inter-subnet routing without knowing physical link details
- Link layer handles local delivery without understanding global routing

Each layer focuses on its specific scope, making the overall system manageable and efficient.

1.1.5 Perspective Matters: Two Views of "Link Layer"

The term "link layer" actually has different meanings depending on your perspective:

IP Subnet Perspective: Link layer moves packets across individual physical links (cable, WiFi connection)

Internet Perspective: Link layer moves packets across entire IP subnets (what we call network layer within a subnet)

From the Internet's viewpoint, each IP subnet is just one "link" in the larger network. This is why the Internet's "link layer" is actually the network layer of individual subnets.

1.2 Link-Layer Services

The link layer (focusing on physical links within subnets) provides several key services:

1.2.1 Error Detection

- Receivers detect and drop corrupted packets using checksums
- Similar to UDP/TCP error detection but at the physical link level

1.2.2 Reliable Data Delivery

- Sender/receiver detect corruption and loss, attempting recovery
- Uses checksums, acknowledgments, and retransmissions
- Typically deployed only on error-prone links (especially wireless)

Why Link-Layer Reliability?

Since TCP provides end-to-end reliability, why also implement it at the link layer? The answer is **performance optimization**.
Consider a long network path where one link experiences frequent packet loss:
Without link-layer reliability:

1. TCP times out waiting for acknowledgments
2. Resets congestion window to minimum
3. Slowly ramps up transmission rate again
4. Overall throughput drops significantly

With link-layer reliability:

1. Link layer locally detects and retransmits lost packets
2. TCP never sees the packet loss
3. No TCP timeout or congestion window reset
4. Maintains higher end-to-end throughput

This local recovery is much faster than end-to-end TCP recovery, especially over long network paths.

1.2.3 Medium Access Control (MAC)

- Manages access to shared physical medium (e.g., wireless spectrum)
- Detects collisions when multiple devices transmit simultaneously
- Implements backoff and retry mechanisms

1.3 Ethernet Networks

Now let's examine how packets actually move within an IP subnet, focusing on Ethernet—the dominant technology for local area networks.

1.3.1 MAC Addresses

Every network interface in an Ethernet subnet has a unique **MAC address** (also called Ethernet address or physical address):

- **Format:** 48-bit number, typically written as six hexadecimal bytes
- **Example:** `5c:f9:38:a4:00:76`
- **Addressing:** Flat (not hierarchical like IP addresses)
- **Scope:** Globally unique but location-independent

Intra-Subnet Communication

When devices communicate within the same IP subnet, packets carry Ethernet headers containing source and destination MAC addresses:

Link-layer header	Network header	Data
src MAC — dst MAC	IP header	Payload

For communication between two devices in the same subnet:

- **Source MAC:** Address of the sending device's network interface
- **Destination MAC:** Address of the receiving device's network interface

Inter-Subnet Communication

When a device sends packets to a different IP subnet, the MAC addressing changes as the packet traverses the local subnet:

- **Source MAC:** Sending device's MAC address
- **Destination MAC:** Border router's MAC address (not the final destination!)

Key principle: Within any IP subnet, packets always carry MAC addresses from devices *within that subnet*. The source MAC belongs to whichever device first forwards the packet in this subnet, while the destination MAC belongs to whichever device will receive the packet last in this subnet. This means MAC addresses change as packets cross subnet boundaries, while IP addresses remain constant end-to-end.

1.3.2 Switch Forwarding

Ethernet switches use **forwarding tables** to decide where to send packets. Each switch:

1. Names its network interfaces (called *links* or *ports*)
2. Maintains a forwarding table mapping MAC addresses to output links
3. For each incoming packet: reads destination MAC address, looks it up, and forwards to the correct output link

L2 vs L3 Forwarding: A Critical Difference

The flat nature of MAC addresses creates a fundamental difference from IP forwarding:

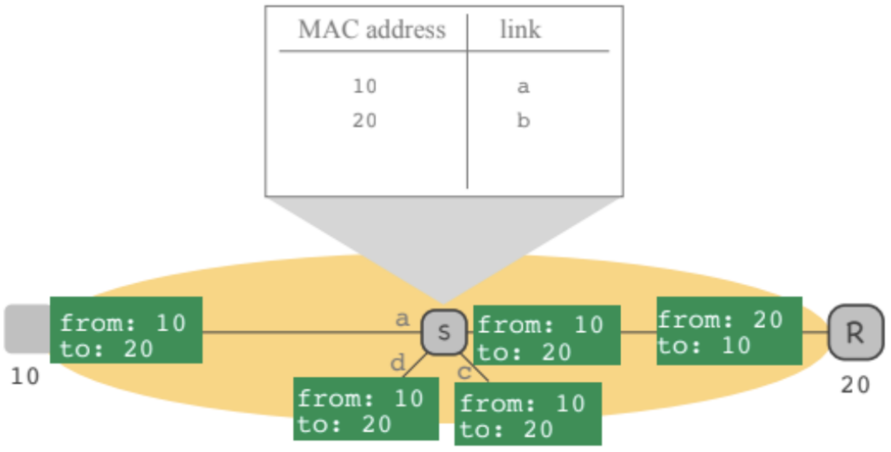
L2 Forwarding (Flat addresses): Cannot group MAC addresses into prefixes; forwarding table size equals the number of active MAC addresses in the subnet

L3 Forwarding (Hierarchical addresses): Groups IP addresses into prefixes; forwarding table size equals local prefixes plus aggregated foreign prefixes

This means Ethernet switches must track every individual device, while IP routers can aggregate millions of addresses into single routing entries.

1.3.3 L2 Learning: Self-Configuring Networks

Unlike IP routers that exchange explicit routing information, Ethernet switches learn automatically from traffic:



Learning Algorithm

1. **Initial state:** Forwarding table is empty
2. **Learning:** When a packet with source MAC address X arrives at link Y, add “MAC X → link Y” to the forwarding table
3. **Unknown destinations:** When a packet arrives with unknown destination MAC, broadcast it to all links

The figure above illustrates this process: when the switch receives a packet from MAC address 10 on link a, it learns that MAC 10 is reachable via link a. Similarly, when it receives traffic from MAC 20 on link b, it updates its forwarding table accordingly.

Learning vs Routing Comparison

L2 Learning: Passive learning from actual traffic; no explicit control messages

IP Routing: Active exchange of routing protocol messages between routers

The Broadcasting Problem: Forwarding Loops

Broadcasting unknown destinations creates a serious problem: **forwarding loops**. If switches naively broadcast to all links, packets can circulate forever in network loops.
Spanning Tree Solution:
To prevent loops, switches use the **Spanning Tree Protocol**:

- Creates a loop-free subgraph connecting all devices
- Includes all nodes but only a subset of links
- Broadcasts propagate only along tree edges
- Eliminates forwarding loops while maintaining connectivity

A spanning tree includes just enough links to reach every device without creating cycles—you cannot remove any edge without disconnecting some node.

1.3.4 Address Resolution Protocol (ARP)

We've explained how switches forward packets using MAC addresses, but a crucial question remains: **How does a device determine the MAC address of its intended recipient?** This is where the Address Resolution Protocol (ARP) comes in.

The Problem: IP to MAC Address Mapping

When Alice wants to send a packet, she faces two requirements:

1. **Destination IP address:** Obtained from DNS (e.g., Bob's IP address)
2. **Destination MAC address:** Unknown—this is what ARP solves

Scenario 1: Same Subnet Communication

When Alice wants to send a packet to Bob in the same IP subnet:

1. **ARP Request:** Alice broadcasts a request asking "Who has IP address 128.178.2.20? Tell 128.178.2.10 (Alice's IP)"
2. **Broadcasting:** Request uses special broadcast MAC address `FF-FF-FF-FF-FF-FF`, reaching every device in the subnet
3. **ARP Response:** Bob recognizes his IP address and responds directly to Alice with his MAC address
4. **Communication:** Alice now knows Bob's MAC address and can send packets directly

Scenario 2: Different Subnet Communication

When Alice wants to send a packet to Bob in a different IP subnet:

1. **Default Gateway:** Alice knows her router's IP address (e.g., 128.178.2.1) through configuration
2. **ARP Request:** Alice broadcasts asking for the router's MAC address
3. **Router Response:** Router responds with its MAC address
4. **Packet Forwarding:** Alice sends packets to Bob using the router's MAC address as destination

ARP vs DNS Comparison

ARP: Uses broadcasting within local subnet; no centralized infrastructure; each device knows its own MAC address

DNS: Uses hierarchical server infrastructure; logically centralized mapping; dedicated servers maintain databases

Both serve address resolution roles but operate at different scales and use different mechanisms.

1.4 Design Trade-offs and Architecture

1.4.1 Could We Eliminate IP Addresses?

An interesting question: Could the entire Internet be one big Ethernet subnet using only MAC addresses?
Answer: No, this wouldn't scale.

- **Forwarding table explosion:** Switches would need individual entries for millions of active devices
- **Broadcasting chaos:** Every address resolution would broadcast to the entire Internet
- **No aggregation:** Flat MAC addresses prevent the hierarchical aggregation that makes IP routing scalable

1.4.2 Could We Eliminate MAC Addresses?

Alternatively: Could we use only IP addresses and routers everywhere?
Answer: Yes, this would scale, but we'd lose flexibility.
The Internet was designed as a *network of networks*—different types of subnets (Ethernet, WiFi, fiber, etc.) interconnected by IP routers. Eliminating the link layer would require replacing all specialized network equipment with IP routers, reducing the diversity and adaptability that make the Internet robust.

1.4.3 Ethernet Elements Summary

The complete Ethernet system relies on three core components:

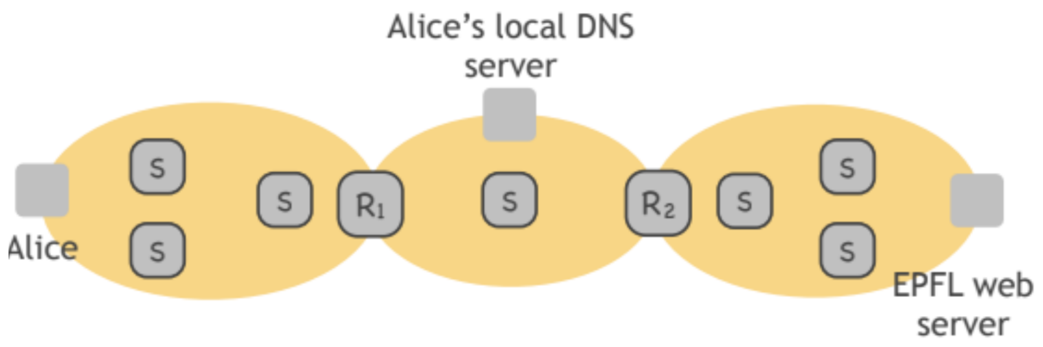
Address Resolution Protocol: Maps IP addresses to MAC addresses (analogous to DNS)

L2 Forwarding: Routes packets based on flat MAC addresses (analogous to IP forwarding)

L2 Learning: Populates switch forwarding tables (analogous to IP routing protocols)

1.5 Complete Example: Packet Journey

Now let's trace through a complete example showing how all these pieces work together when Alice sends a DNS request to access a web server:



1.5.1 The Scenario

When Alice types `http://www.epfl.ch` in her browser, this generates at least four packets:

1. Alice's DNS request to local DNS server
2. Local DNS server's response to Alice
3. Alice's HTTP GET request to web server
4. Web server's response to Alice

We'll focus on the first packet: Alice's DNS request to her local DNS server.

1.5.2 Step-by-Step Packet Journey

Step 1: Application Layer Creates DNS Request

Alice's DNS client process creates a DNS request to resolve `www.epfl.ch`, which is passed down to the transport and network layers:

- **Source IP:** Alice's IP address
- **Destination IP:** Local DNS server's IP address (known via configuration)

Step 2: ARP Request for Local Router

Alice's network layer needs to determine the appropriate destination MAC address. Since the DNS server is in a different subnet, Alice must send the packet to her default gateway (router R1). She broadcasts an ARP request to resolve the router's IP address to its MAC address.

Step 3: ARP Response from Router

Router R1 receives the ARP request, recognizes its own IP address, and responds with its MAC address.

Step 4: DNS Request with MAC Headers

Alice can now send the DNS request with proper addressing:

- **Source MAC:** Alice's MAC address
- **Destination MAC:** R1's MAC address
- **Source IP:** Alice's IP address
- **Destination IP:** DNS server's IP address

Step 5: Router Performs IP Forwarding

R1 receives the packet, examines the destination IP address, and consults its forwarding table to determine the next hop toward the DNS server.

Step 6: Router's ARP Request

R1 needs to forward the packet to the next subnet containing the DNS server. It broadcasts an ARP request to resolve the DNS server's IP address to its MAC address.

Step 7: DNS Server's ARP Response

The DNS server receives the ARP request and responds with its MAC address.

Step 8: Final Packet Delivery

R1 forwards the DNS request with updated MAC addresses:

- **Source MAC:** R1's MAC address (in the new subnet)
- **Destination MAC:** DNS server's MAC address
- **Source IP:** Alice's IP address (unchanged)
- **Destination IP:** DNS server's IP address (unchanged)

Address Requirements

- **End-systems and routers need MAC addresses:** Otherwise, switches wouldn't know where to forward packets within subnets
- **End-systems need IP addresses:** Otherwise, routers wouldn't know where to forward packets across the Internet
- **Switches don't need MAC addresses for forwarding:** They learn MAC-to-port mappings automatically
- **Routers don't need IP addresses for forwarding:** They use destination IP addresses in packet headers

1.6 Network Hierarchy Summary

The Internet operates at three distinct levels, each with its own addressing and routing mechanisms:

1.6.1 Level 1: IP Subnets

- **Devices:** Link-layer switches connecting end-systems and routers
- **Forwarding:** L2 forwarding based on MAC addresses
- **Learning:** L2 learning populates switch forwarding tables automatically

1.6.2 Level 2: Autonomous Systems (AS)

- **Devices:** IP routers connecting multiple IP subnets within one administrative domain
- **Forwarding:** IP (L3) forwarding based on IP addresses
- **Routing:** Intra-domain routing protocols (OSPF, RIP) populate router forwarding tables

1.6.3 Level 3: Internet

- **Devices:** Border routers connecting multiple Autonomous Systems
- **Forwarding:** IP (L3) forwarding based on IP addresses
- **Routing:** Inter-domain routing protocol (BGP) coordinates between ASes

Each level uses appropriate addressing schemes and protocols optimized for its scale and administrative requirements, creating a hierarchical system that enables the Internet's remarkable scalability and robustness.