Multi-Hop Heterogeneous Networking (1/3)

- Interconnecting multiple LANs that employ the same MAC may be achieved by bridging protocols, applying MAC layer addresses.
- *Heterogeneous Internetworks* consist of many smaller networks (e.g., LANs) with different MACs.
- Interconnecting multiple LANs with different MAC implementations require packet forwarding based on layer 3 addresses ⇒ *Internet Protocol (IP)*
  - Connectionless datagram delivery service
  - Send packet with IP address of destination in the datagram header to best known next hop and “hope for the best”
  - Time-to-Live (TTL) or Hop Limit field ensures IP packet eventually expires (or purged) if never delivered.
Multi-Hop Heterogeneous Networking (2/3)

Example: Host 1 and Host 4 exchange files (over TCP/IP):

- A: Ethernet
- B: Serial, point-to-point link
- C: Token Ring
- D: Ethernet

```
A: Ethernet
B: Serial, point-to-point link
C: Token Ring
D: Ethernet
```

Example:

Host 1 and Host 4 exchange files (over TCP/IP):

```
Host 1
TCP
IP
Ethernet

Router 1
IP
Ethernet
PPP / Serial

Router 2
IP
PPP / Serial
Token Ring

Host 4
TCP
IP
Token Ring
```

IP addressing permits end-to-end packet forwarding

Multi-Hop Heterogeneous Networking (3/3)

- Devices known as routers perform packet forwarding between heterogeneous LANs based on forwarding tables to ranges of IP addresses
  - Forwarding towards a range of addresses summarizes topology detail (i.e., routing information)
  - Forwarding tables constructed at routers via some routing protocol for which routers exchange messages (recall examples of routing algorithms in Ch. 5)
  - Originating host forwards frames for a non-local destination to a known gateway router
- Packet delivery across a link requires mapping of an IP address to a link layer address
  - Link layer (i.e., layer 2) frame encapsulates IP (i.e., layer 3) datagram
  - \( \Rightarrow \) Address Resolution Protocol (ARP)
IP Packet Format (1/7)

- The Internet Protocol (IP) was originally invented in the 1970s
- IP provides best-effort connectionless delivery of datagrams
- Effectively hides link layer implementation from higher layer protocols
- Supports interoperability across diverse link layer technologies
- Several versions have been developed
  - E.g., IPv4: Current de facto standard
  - E.g., IPv6: Next generation IP standard

IP Packet Format (2/7)

- IPv4 (IP version 4) packet format:

```
| 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Version | HLen | Type of Service | Length |
| Identification | Flags | Offset |
| Time-to-live (TTL) | Protocol | Checksum |
| Source Address |
| Destination Address |
| Options (variable) | Pad (variable) |
| Data (variable) |
```
IP Packet Format (3/7)

- IPv4 packet format (continued)
  - **Version**: Indicates version of IP that generated the packet header (e.g., 4)
  - **Internet Header Length (IHL)**: Specifies length of header in terms of 32-bit words
  - **Type of Service (ToS)**: Specifies the quality of service requested by the datagram source
  - **Length**: Total length of the IP datagram in bytes (includes header)

IP Packet Format (4/7)

- IPv4 packet format (continued)
  - **Identification**: This number along with the Protocol, Source and Destination fields uniquely identifies datagram, allowing unambiguous fragmentation and reassembly
  - **Flags**: 2 bits are used fragmentation and reassembly and a third is unused
  - **Offset**: Specifies the distance in 8-byte (64-bit) blocks that a fragment is offset from the first byte (bit) of the original datagram
IP Packet Format (5/7)

- **IPv4 packet format (continued)**
  - **Time-to-live (TTL):** Specifies the duration that a datagram is allowed to persist in the network prior to deliver to the destination host
  - **Protocol:** Specifies the identity of the higher layer protocol that originated the datagram (e.g., TCP, UDP)
  - **Checksum:** 16-bit checksum computed on the IP header (only) via ones complement arithmetic

<table>
<thead>
<tr>
<th>Time-to-live (TTL)</th>
<th>Protocol</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

IP Packet Format (6/7)

- **IPv4 packet format (continued)**
  - **Source Address:** Specifies the IP address of the interface connected to the host that originated the datagram
  - **Destination Address:** Specifies the IP address of an interface connected to the host that is the intended recipient of the datagram

<table>
<thead>
<tr>
<th>Source Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31</td>
</tr>
<tr>
<td>Destination Address</td>
</tr>
</tbody>
</table>
IP Packet Format (7/7)

<table>
<thead>
<tr>
<th>Options (variable)</th>
<th>Pad (variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Data (variable)</td>
<td></td>
</tr>
</tbody>
</table>

- IPv4 packet format (continued)
  - **Options**: Specifies options such as security and source routing
  - **Pad**: Ensures that the header ends on a 32-bit boundary
  - **Data**: Message payload from higher-layer protocol

Fragmentation and Reassembly (1/7)

- Since IP is designed to operate over all types of link layer configurations, IP must support packet forwarding over interfaces with a wide range of maximum and minimum permissible packet lengths
  - **Maximum transmission unit (MTU)**: Largest IP datagram that fits into the payload of a layer 2 protocol frame
    - MTU may vary for each type of layer 2 protocol depending on factors such as fairness (in multiple access schemes) and frame error rates (longer frames are more prone to errors)
    - E.g., Ethernet: MTU = 1500 bytes
Fragmentation and Reassembly (2/7)

- E.g., suppose a datagram originated at source host 1 must traverse 3 networks as follows:
  1. Ethernet with MTU = 1500 bytes
  2. Serial link implemented with MTU = 640 bytes
  3. Token ring implemented with MTU = 4464 bytes

If the original datagram consists of 1300 bytes of layer 4 payload plus 20 bytes of IP header, how can datagram forwarding be achieved?
- IPv4 implements a function known as **fragmentation** to partition original datagram into multiple datagrams of smaller size that fit the MTU
- Destination performs **reassembly** of datagram fragments to regenerate original datagram

Fragmentation and Reassembly (3/7)

Example: Host 1 sends IP datagram with 1300 byte payload (and 20 byte header) to host 4:

A: Ethernet — MTU = 1500 bytes
B: Serial, point-to-point link — MTU = 640 bytes
C: Token Ring — MTU = 4464 bytes
D: Ethernet

Total datagram length is 1320 bytes → How is the datagram fragmented at router R1 for transmission over serial link B?
Fragmentation and Reassembly (4/7)

- **IPv4 fields to support fragmentation:**
  - **Identification:** Allows fragments of same datagram to be identified at destination
  - **Offset:** Specifies the distance in units of 8-byte blocks from the start of the original datagram payload to the start of the fragment payload (e.g., for 1\textsuperscript{st} fragment, **Offset = 0**)
  - **More Flag:** 3\textsuperscript{rd} bit of 3-bit flag field, indicating whether the fragment is the last fragment (i.e., set to 0) of the original datagram or if more fragments are to follow (i.e., set to 1)

Fragmentation and Reassembly (5/7)

For transmission from host 1 to router R1 on link A (Ethernet with MTU = 1500 bytes), no fragmentation is required
For transmission from router R1 to router R2 on link B (serial link with MTU = 640 bytes), fragmentation into 3 datagrams is required

- **Fragment 1**
  - Length = 636
  - Identification = \_\_\_\_\_\_\_\_\_\_\_\_
  - Offset = 0
  - Data (616 data bytes)

- **Fragment 2**
  - Length = 636
  - Identification = \_\_\_\_\_\_\_\_\_\_\_\_
  - Offset = 77
  - Data (616 data bytes)

- **Fragment 3**
  - Length = 88
  - Identification = \_\_\_\_\_\_\_\_\_\_\_\_
  - Offset = 154
  - Data (68 data bytes)

Further fragmentation is **not** required on link C (Token Ring interface) — Why?

**Fragmentation and Reassembly (7/7)**

- Destination host 4 receives 3 IP datagram fragments on link C (token ring interface)
- Must receive all fragments in order to reassemble the original datagram
- Drawbacks of fragmentation/reassembly
  - Extra processing delay at routers to do fragmentation
  - Even if only one fragment is lost or received in error, entire original datagram must be retransmitted