



The Internet Protocol

Jeffrey Shafer



Why not just use Ethernet?

- Most computer systems use Ethernet networking
- Ethernet provides facilities to
 - Locate computers
 - Forward packets directly
 - Prevent loops
 - ...
- What are the drawbacks of Ethernet for global communication?



Ethernet Drawbacks

- Locating computers

- Do we really want to broadcast across the Internet?

- Preventing loops

- Do we really want to rebuild an Internet-wide spanning tree whenever the topology changes?
- Do we really want packets to live forever if loops remain?

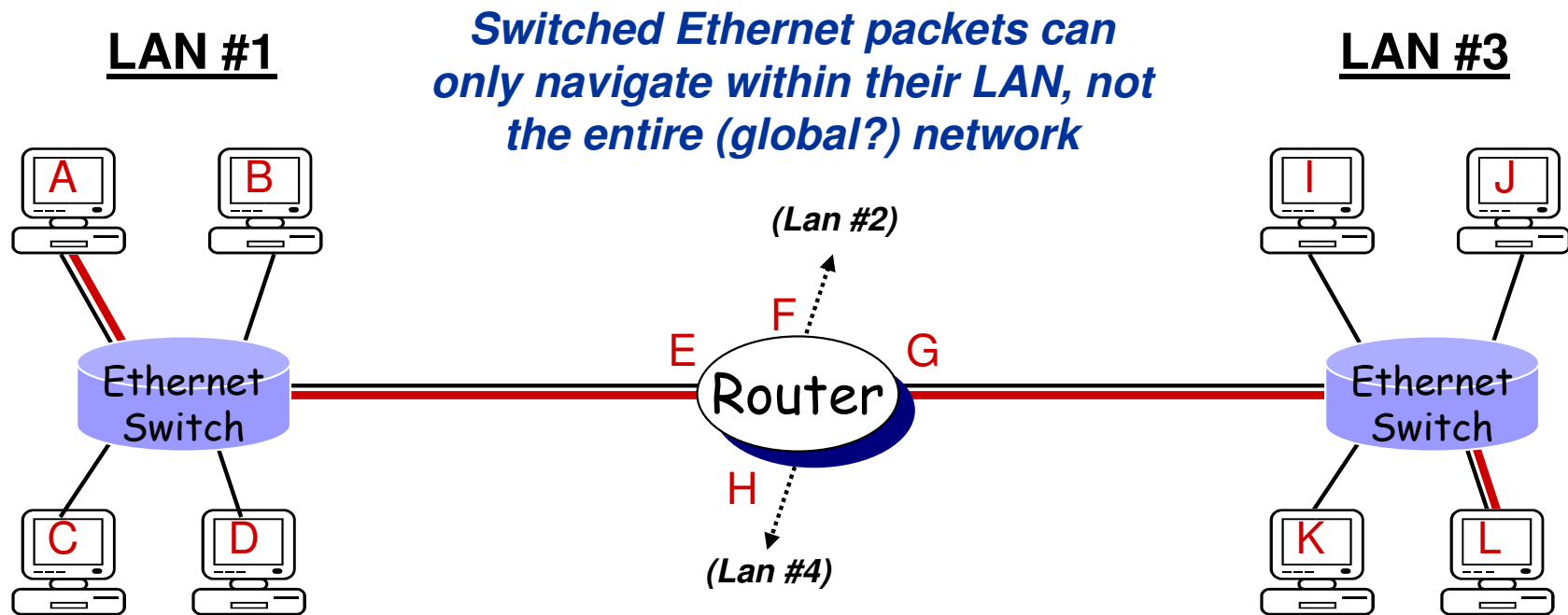
- Unreachable computers

- What happens if the destination is unreachable?
- I.e., it doesn't exist, is turned off, is broken, ...

Network vs. Link Layer

- “Most” ≠ “All”
 - Not all networks are Ethernet
 - Why limit choice, innovation, etc. at the link level?
- Link layer
 - What is the best way to move local traffic (single hop)?
 - Old/new network, wired/wireless network, ...
 - Different links can use different networks!
- Network layer
 - What is the best to handle multi-hop communication?
 - Addressing
 - Unreliable delivery mechanisms (routing)

Routing Between LANs



(1) A transmits to L using higher-level protocol (e.g. IP)
Ethernet frame destination is router

Frame:

DA (E)	SA (A)	Type / Data	CRC
--------	--------	-------------	-----

(2) Switch forwards frame to router

(3) Router uses higher-level protocol to determine destination, and updates Ethernet frame destination, source and CRC

Frame:

DA (L)	SA (G)	Type / Data	CRC
--------	--------	-------------	-----

(4) Switch forwards frame to destination



The Internet Protocol

- **Datagram**

- Each packet is individually routed
- Packets may be fragmented or duplicated
 - Due to underlying networks

- **Connectionless**

- No guarantee of delivery in sequence

- **Unreliable**

- No guarantee of delivery
- No guarantee of integrity of data

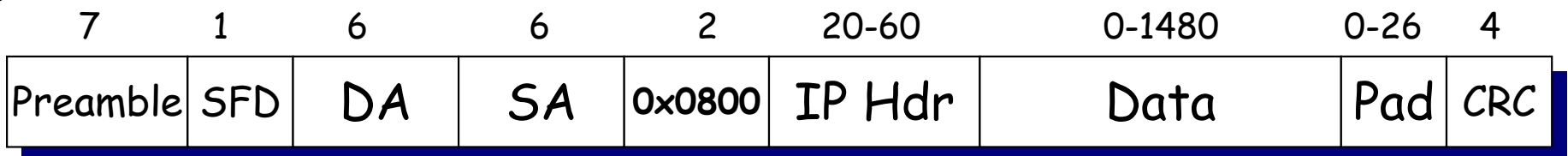
- **Best effort**

- Only drop packets when necessary
- No time guarantee for delivery

IP and Ethernet

- IP datagrams can be *encapsulated* in Ethernet frames

Bytes:



IP Datagram



Understanding IP

- Datagram lifetimes
 - Time-to-live
- Handling disparate link layers
 - Fragmentation
- IP integrity
 - Header checksum
- What do datagrams look like?
 - Header format
- Addressing

Time-to-Live

- Sender sets a TTL value for each datagram
- Each router decrements the TTL
- When the TTL reaches 0
 - The router drops the datagram
 - The router sends an ICMP error (more later) to the sender
- Effectively a “maximum hop count”
- Why is this useful/interesting/necessary?



Traceroute

- Tool to find the route IP packets take through the Internet
- Exploits the TTL field
 - Send packets with successively increasing TTL values
 - See what routers respond with ICMP “Time Exceeded” errors

Using Traceroute

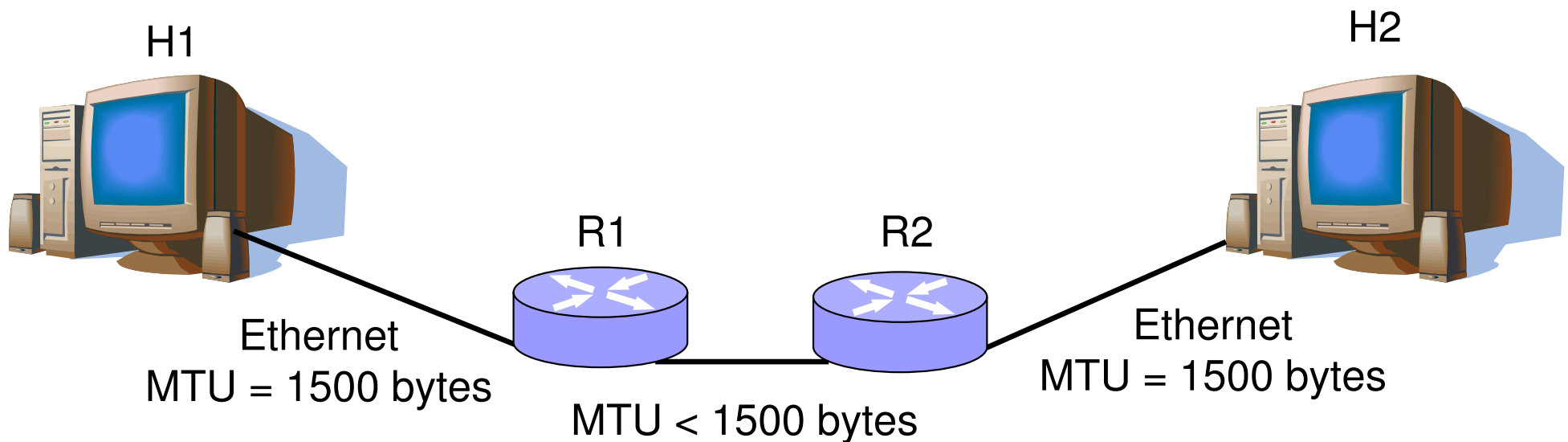
```
rixner@comp519:~$ traceroute www.rice.edu
traceroute to www.rice.edu (128.42.206.11), 30 hops max, 40 byte
  packets
 1  staff-74-dun20-254.rice.edu (10.74.20.254)  0.382 ms  0.429
    ms  0.493 ms
 2  b48-c1-b20-d1.net.rice.edu (172.16.20.245)  0.868 ms  0.858
    ms  0.860 ms
 3  172.16.0.246 (172.16.0.246)  0.853 ms  0.849 ms  0.843 ms
 4  128.42.206.254 (128.42.206.254)  1.014 ms  1.076 ms  1.164 ms
 5  128.42.206.11 (128.42.206.11)  0.400 ms  0.398 ms  0.392 ms
rixner@comp519:~$
```

TTL Value

Router name/IP address

Round trip time of 3 probes

The Maximum Transmission Unit



- Not all networks have the same maximum transmission unit (MTU)
- IP datagrams can be larger (64KB) than the Ethernet MTU (1500 bytes)
- Would like IP datagrams to still get delivered!

Fragmentation

- Routers fragment IP datagrams
 - Break datagram into MTU-sized datagrams
 - Set “MF” (more fragments) flag, as necessary
 - Set fragment offset field appropriately
- May need to fragment already fragmented datagrams!
- Datagrams are reassembled by the receiver
 - Routers do not reassemble fragments
 - Once fragmented, a datagram remains fragmented



Routing Fragments

- Fragments are routed independently through the network
- Fragment could be lost or duplicated
- Receiver must reassemble fragments
 - May arrive out of order
 - May not all arrive – must drop entire datagram

Avoiding Fragmentation

- Source can prevent fragmentation with “DF” (don’t fragment) flag
 - Routers send “ICMP” (more later) error to the sender if packet is too large for link MTU
- Hosts commonly try to avoid fragmentation
 - Use “path MTU Discovery” to find the smallest MTU on the path
 - Repeatedly send successively smaller datagrams until no fragmentation occurs
 - Try it yourself with “tracert -F *host size*”

IP Checksum

- Allows hosts/routers to detect corruption in the IP header
 - Payload is not covered
 - Not enough information to correct errors
- 16-bit one's complement checksum
 - Set checksum field in header to zero
 - One's complement sum of all 16-bit values (use end-around carry) and negate (one's complement) result
- Does this provide reliability?
 - Why is it there?

Calculating the IP Checksum

```
unsigned short cksum(uint16 *ip, int len) {
    uint32 sum = 0;

    while (len > 1) {
        sum += *ip++;
        if(sum & 0x80000000) /* if high order bit set, fold */
            sum = (sum & 0xFFFF) + (sum >> 16);
        len -= 2;
    }

    if (len) /* take care of left over byte */
        sum += (uint16) *(uint8 *)ip;

    while (sum >> 16)
        sum = (sum & 0xFFFF) + (sum >> 16);

    return ~sum;
}
```

Verifying the IP Checksum

- Two methods
 - Recalculate IP checksum (with 0 in checksum header field) and compare to checksum in header
 - One's complement sum of entire IP header (including checksum) and compare to -0 (0xffff)
- Why does the second technique work?
- Which is better?

Incremental Update of Checksum

- Routers must modify the TTL of *every* IP datagram
- Checksum must be recomputed!
- How do routers recompute the checksum?
 - From scratch?
 - Incremental update?
- Incremental update:

$$\begin{aligned} HC' &= \sim (C + (-m) + m') && \text{16-bit sum of header} \\ &= \sim (\sim HC + \sim m + m') && \text{Header checksum} \end{aligned}$$



Next Time

- IP Addressing