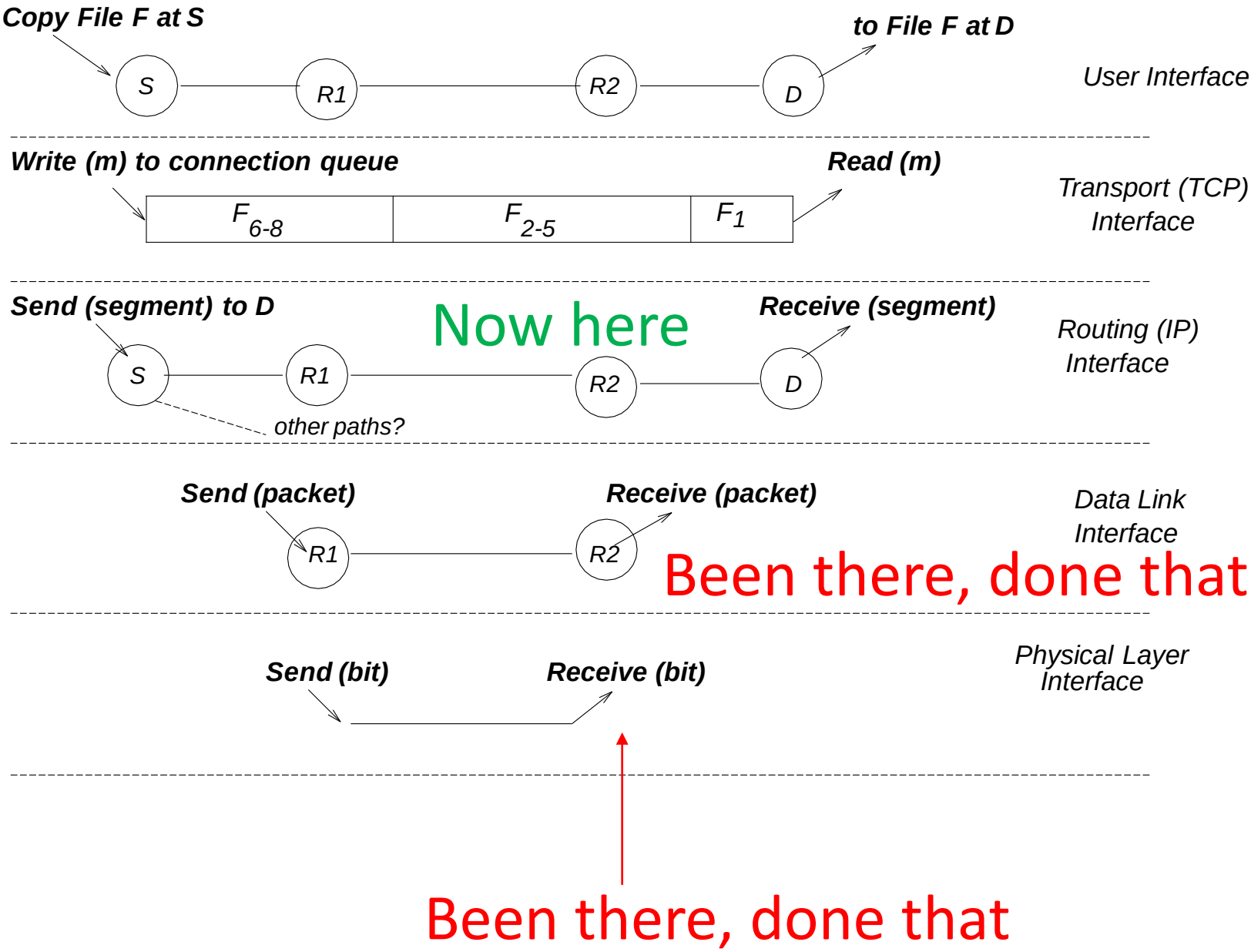
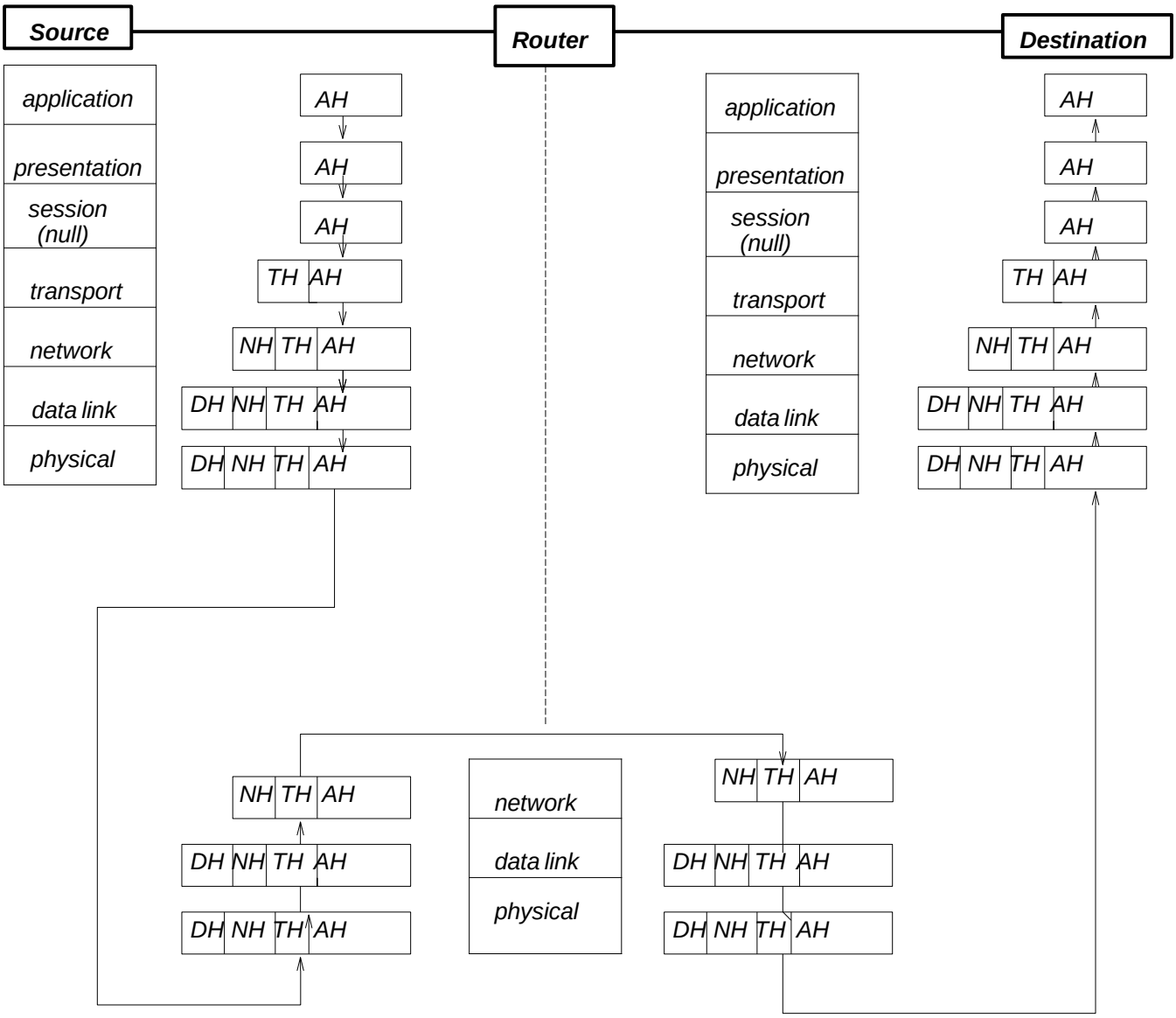
CS 118: IP Addressing and Overview and Project Intro

George Varghese



RECALL THE IP ABSTRACTIONS:

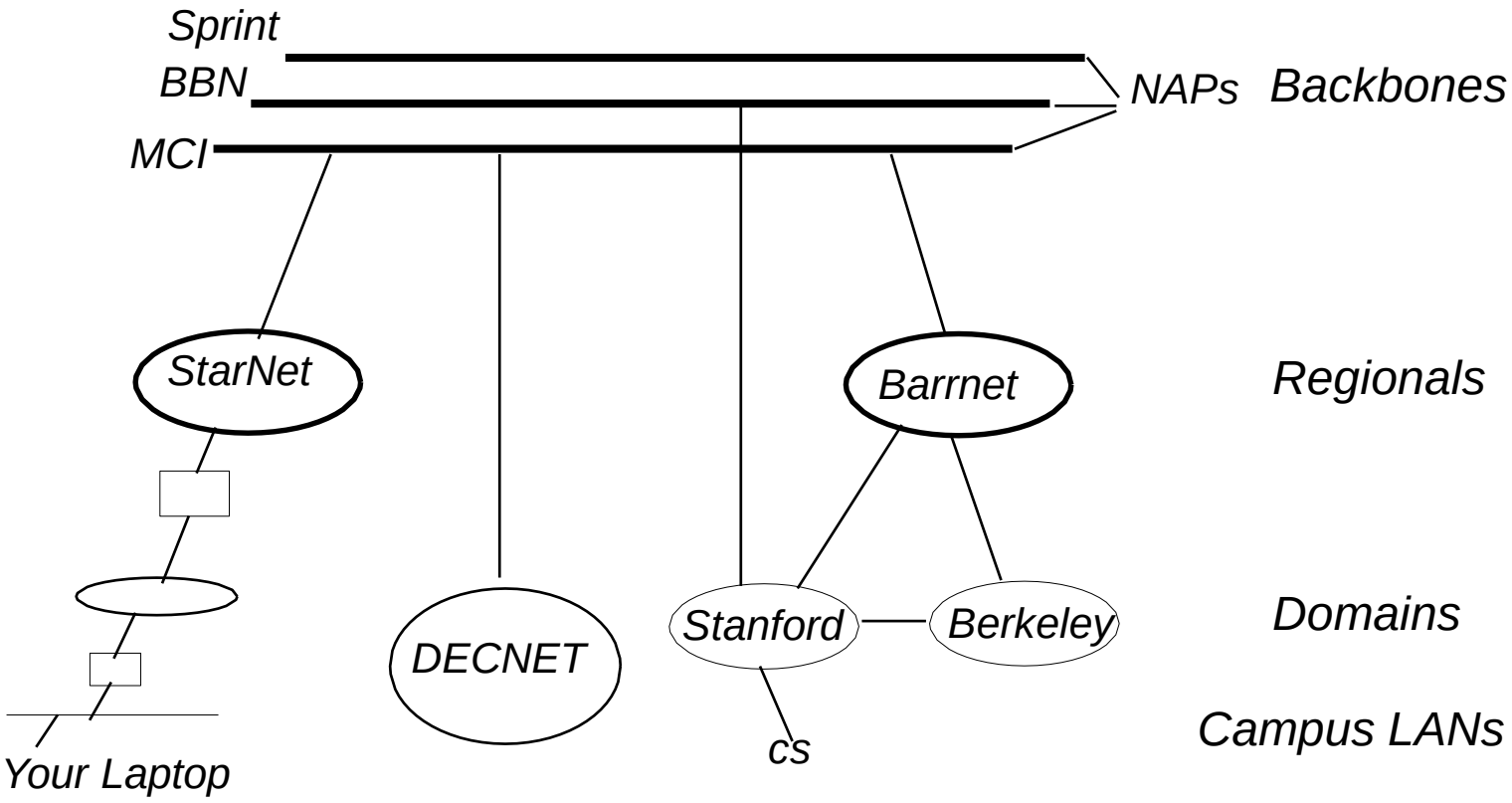


HEADERS: WATCH HOW THEY ARE ADDED AND REMOVED. STRICT LAYERING – THE DEFINITION

Life through the the Internet contains and endless set of unsorted pings. These prompts demand our attention, and present us with a perpetual onslaught of distractions and diversions . . . Our lives can easily be the story of how we respond to this endless set of prompts

From *Called*, by Mark Labberton, president of Fuller Seminary

TOPOLOGY



- Terminology: ISPs, POPs, Autonomous Systems, NAPs, Peering

Basic Internetworking in IP

- *Goal from start* Unlike DECNET, SNA etc. starts with a hierarchy of physical networks with network specific routing that IP does not care about, to create an Internetwork of physical networks
- *IP 's role* to route packets to the right physical network based on the network number. Offers a so-called datagram service with possible fragmentation and reassembly to deal with different maximum packet sizes
- *Error messages* companion protocol called ICMP for error messages (header checksum failed, maximum time exceeded, redirect etc.).

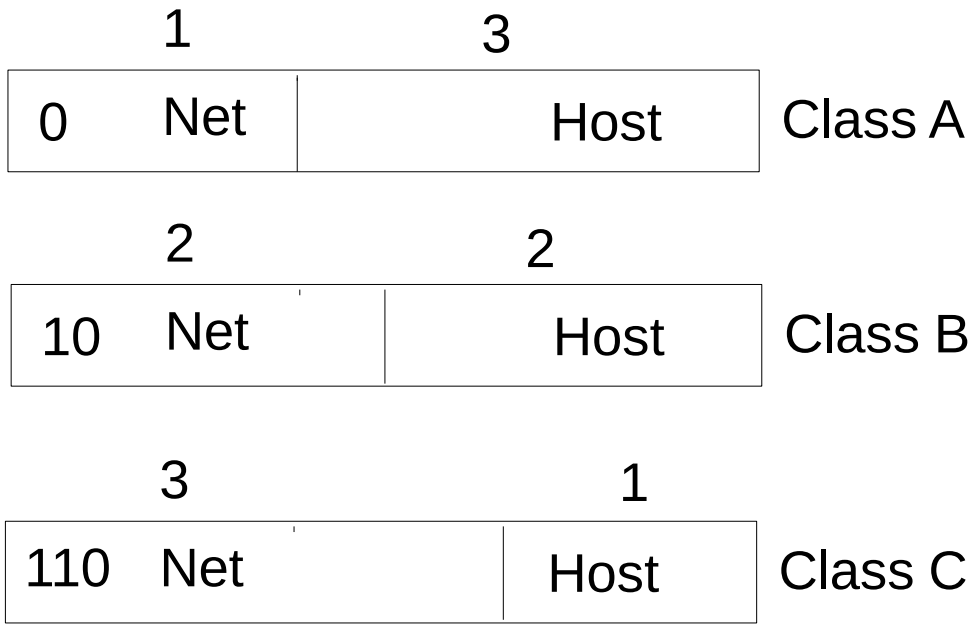
IP Evolution

- *ARPANET* started by linking government and university sites (including UCLA) in the 1970s
- *NSFNet* in 1983 ARPANET splits up into MILNET and ARPANET. In 1984 NSF establishes NSFNET to be backbone. Campuses attached to backbone via regional networks (NYSERNET etc.) Strict hierarchy breaks down because of direct connections between providers
- *Multiple Providers by lat 1980s* Internet becomes worldwide. From a research network to production quality. Multiple autonomous providers that need to work toi=gether

Names and Address

- *Names* when you send to a domain name like cs.Berkeley.edu , a resolver is your host translates the name to a 32-bit IP address. All messages carry IP destinations addresses
- *Domain Name Service* the translation is done using the so-called Domain Name Service or DNS which we will study later

ORIGINAL IP ADDRESSES: CLASSFUL



- Original Model: Small number of large networks (class A), moderate number of campus networks, large number of LANs
- Idea: Hierarchical address with a moverable boundary

Old IP Forwarding

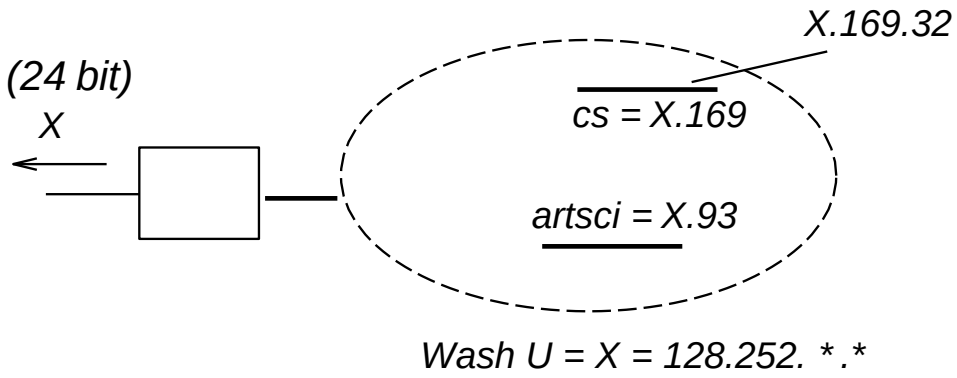
- *Find Destination* extract Network Number of destination address by parsing and checking for class A, class B etc,
- *Final hop reached?* If (Network Number of Dest = Network number of one of this router's local interfaces) deliver packet. Map to local address using ARP or some such network specific protocol
- *Lookup Router Table* Lookup Network Number in the corresponding routing table, If it exists, deliver packet to corresponding NextHop.
- *Lookup Router Table:* If no route entry exists, send yo default router. (This looks silly but is a great way to avoid keeping lots of table entries in stub organizations like UCLA).

Challenge-Response

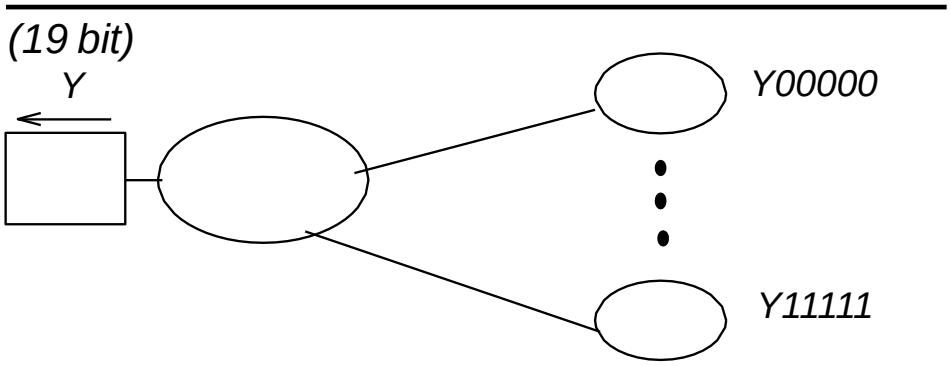
One level of hierarchy good but IP quickly ran into two scaling challenges:

- *Inefficient address usage:* any organization that needed more than 255 addresses asked for a class B address (64,000) and they quickly ran out
- *Routing Table Growth:* the response to no more class B addresses was to assign multiple Class C addresses. But now every backbone router needed to know more addresses, more routing traffic, search times etc.
- *Response* changed IP forwarding to longest matching prefix. Why?

SUBNETTING AND SUPERNETTING



Subnetting a Class B address X



Supernetting Class C addresses Y0−Y31

- Supernetting: Done recursively, leads to backbone routers only having hundreds of thousands of prefixes of lengths 8-32
- Temporary Measures: Often today new organizations are give 1 IP address and use NAT. Need the move to IPV6 (128 bits)

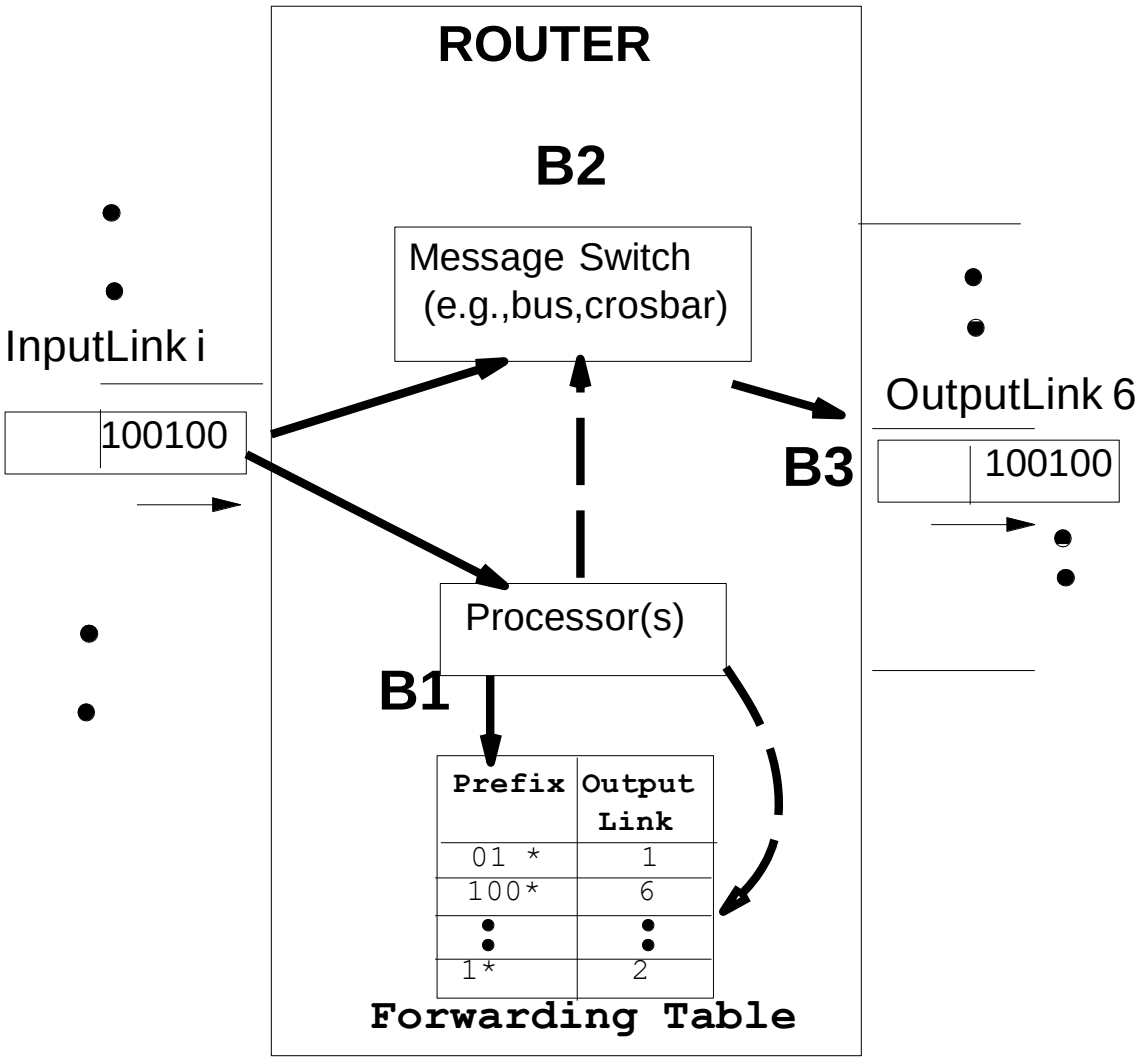
IP evolved to meet challenges

- *Challenge 1* Interconnecting diverse networks → Net Numbers (Class A, 1 byte)
- *Challenge 2:* Ethernets led to an explosion of networks → Hack to add Class B, Class C
- *Challenge 3* Class B addresses ran out → Give consecutive class C and use Longest Prefix Match
- *Challenge 4*: Even Class C's started running out → NAT and concurrent move to IP v6

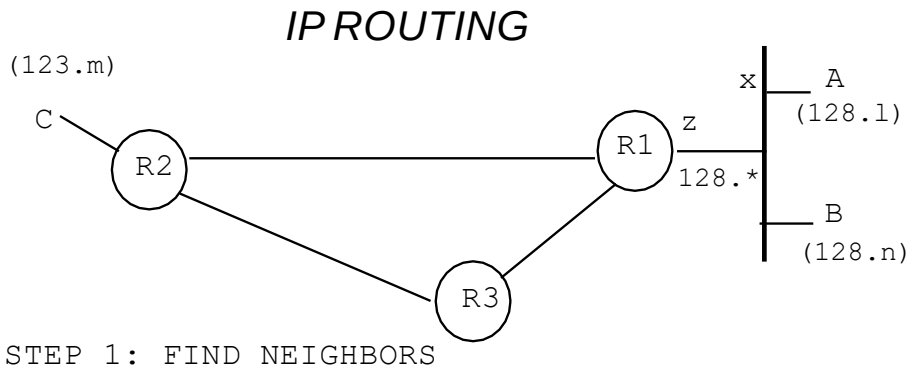
New IP Forwarding

- *Lookup* find longest matching prefix P of destination IP address in packet in forwarding table
- *Default or Local?* If P is nil forward on default route. If the next hop associated with P is a local interfaces, deliver packet. Map to local address using ARP or some such network specific protocol
- *Send on its way* if not, send packet to NextHop route associated with P
- Backbone routers in default-free zone have to have many hundred thousand prefixes to reach everyone. Enterprise routers have 1000s because of heavy use of default routes.

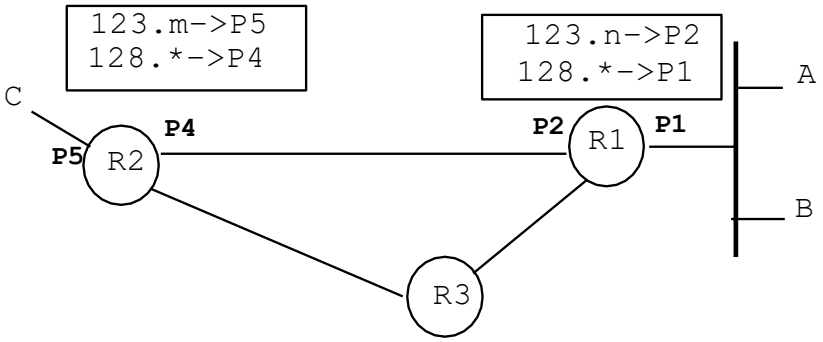
ROUTER MODEL



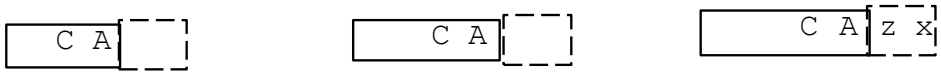
FORWARDING AND ROUTING



STEP 1: FIND NEIGHBORS

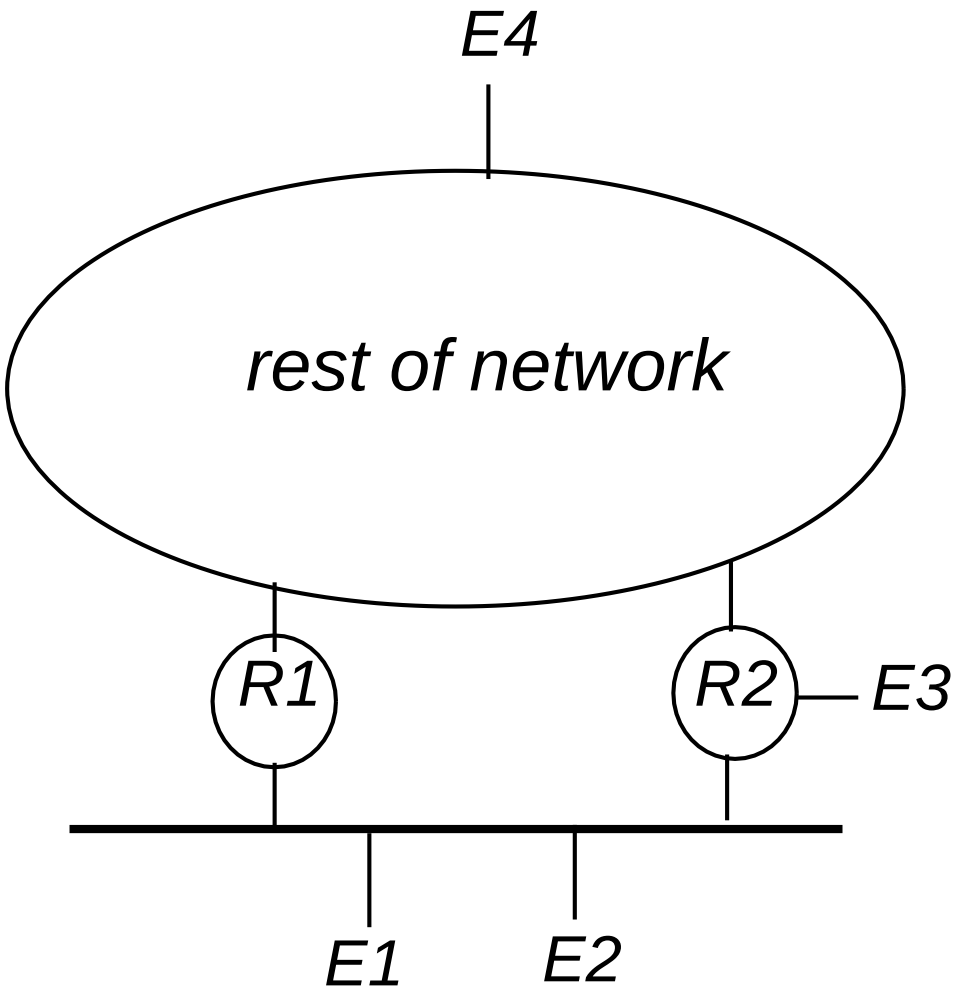


STEP 2: COMPUTE ROUTES



STEP 3: FORWARD

FOUR PROBLEMS ENDNODES MUST SOLVE

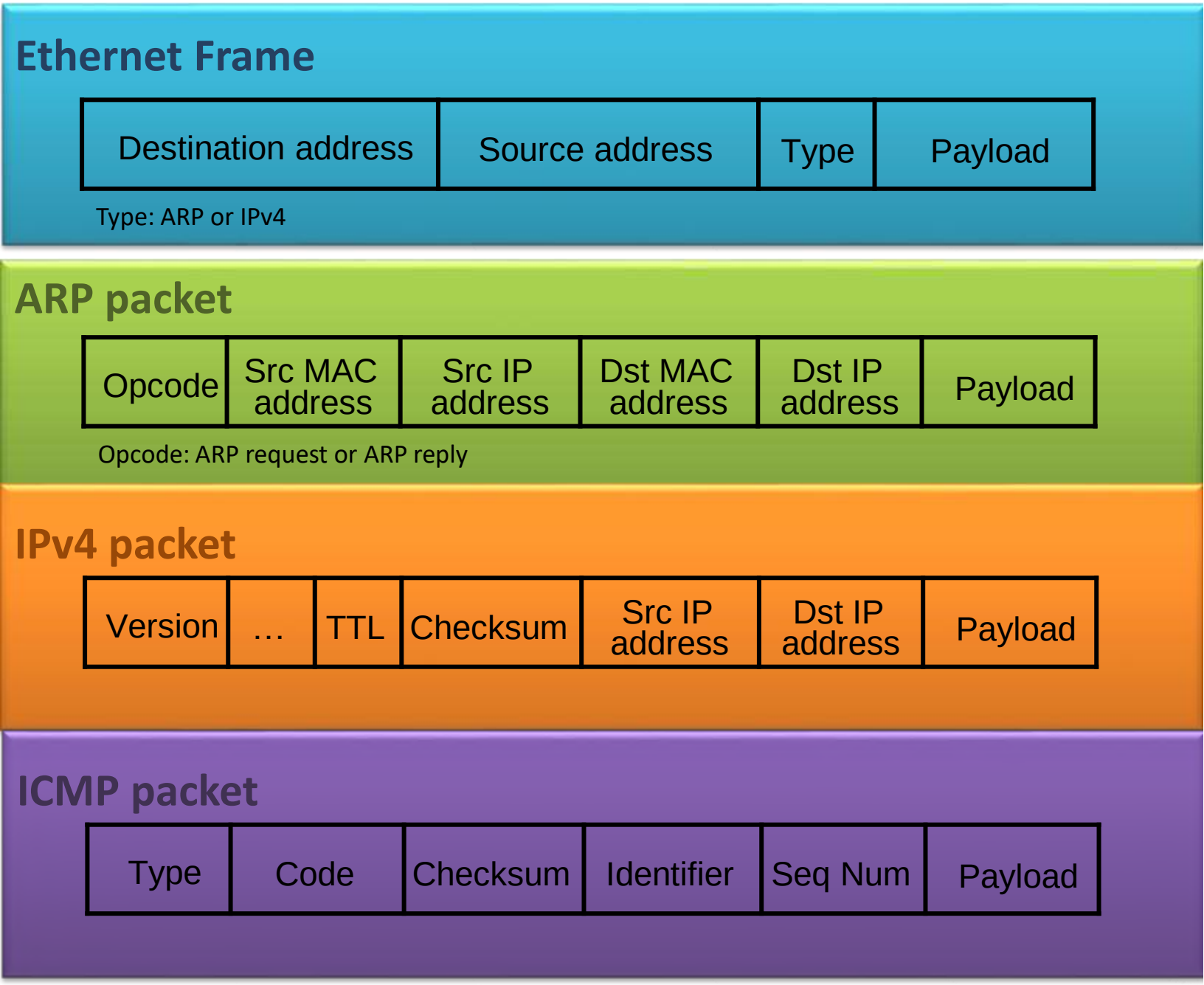


- P1: Routers need Data Link Addresses of endnodes
- P2: Endnodes need DL address of 1 router
- P3: E1 and E2 should be able to communicate without a router
- P4: E1 to E3 traffic should go through R2

IP Solutions to End-node Problems

- *P1:* ARP for MAC address of destination
- *P2:* a service called called DHCP gives you the IP address of one router (autoconfiguration)
- *P3:* two endnodes know they are on same subnet by comparing masks. Then ARP
- *P4:* send to router and router sends redirect if packet returns on interface it entered router. (Ignore this code in project),

Packet Format



IPv4 header also contains header length, total length, ID, flags, fragment offset, and protocol fields.

Forwarding Pseudocode: ARP

- 1. Find input network interface: findIfaceByName. Drop packet if interface is unknown

2. Read ethernet header and check the eth_type field. Ignore all but ARP and IPv4 types

3. If eth_type is ARP:

a. If ARP Request packet:

- Prepare and send ARP response packet

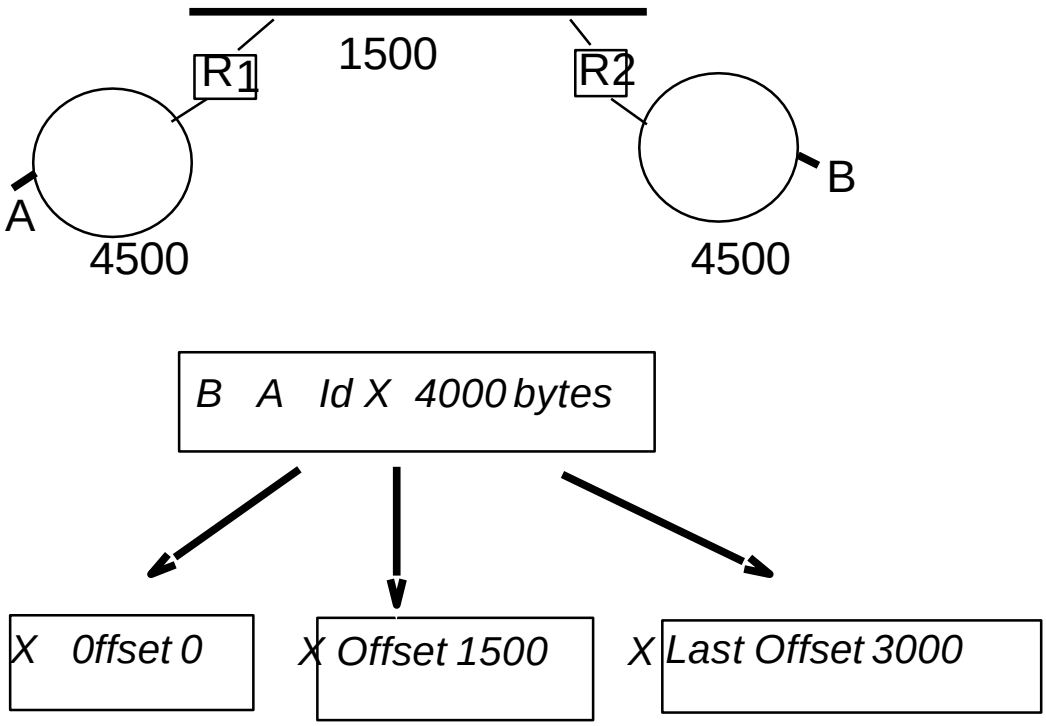
b. If ARP Response packet:

- record IP-MAC mapping information in ARP cache
- send out all enqueued packets for ARP entry

Forwarding Code: IPv4

- 4. If eth_type is IPv4:
  - verify checksum, length, discard invalid packets
  - if packet is to router
    → If ICMP packet then handle Ping
- 5. Use the Longest Prefix Match algorithm to find a next-hop IP address in the routing table
- 6. Lookup ARP cache for MAC address mapped to the next hop destination IP address
  -- If valid entry found: forward packet
  - Else: queue received packet and send ARP request to discover the IP-MAC mapping.

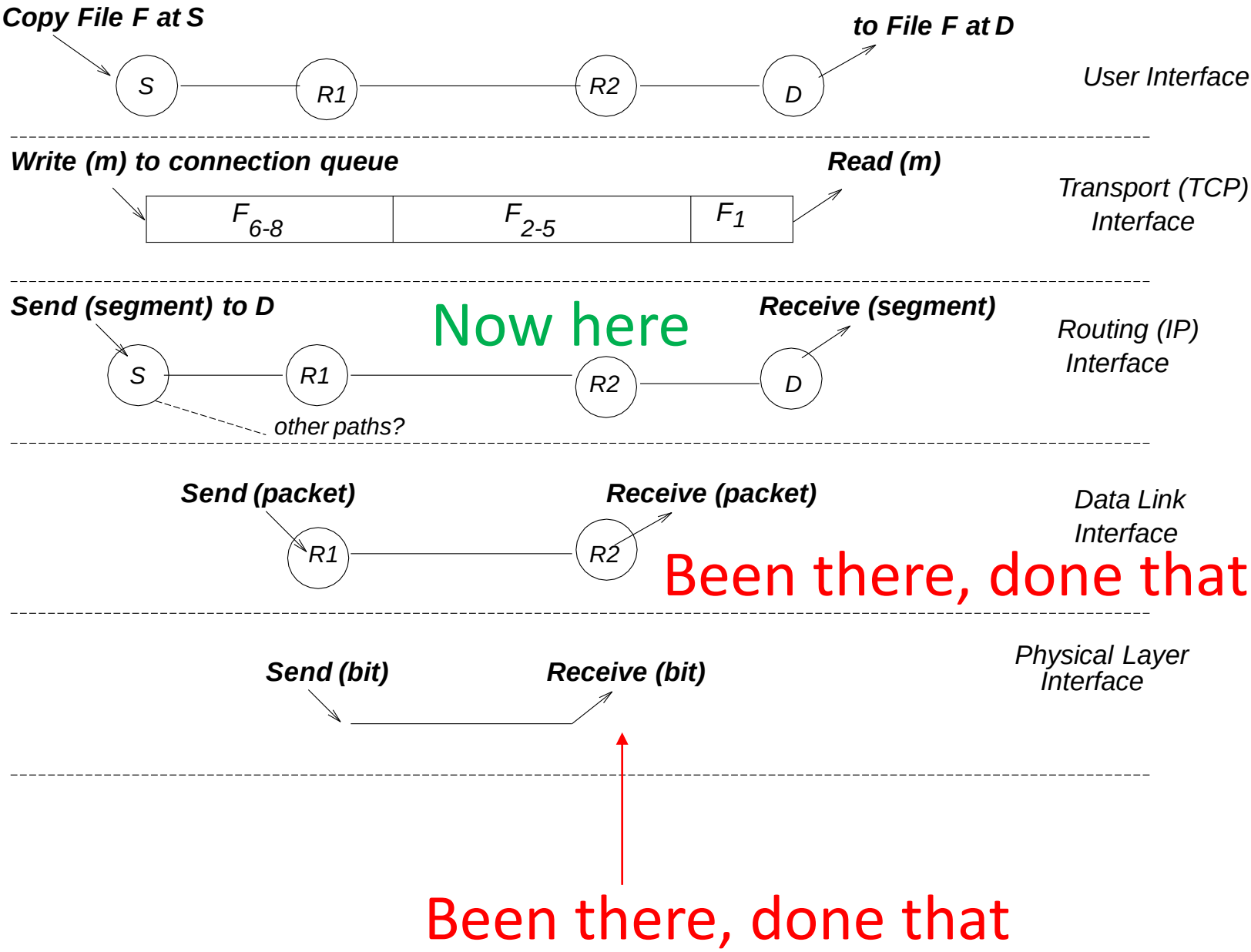
ASIDE ON FRAGMENTATION AND REASSEMBLY



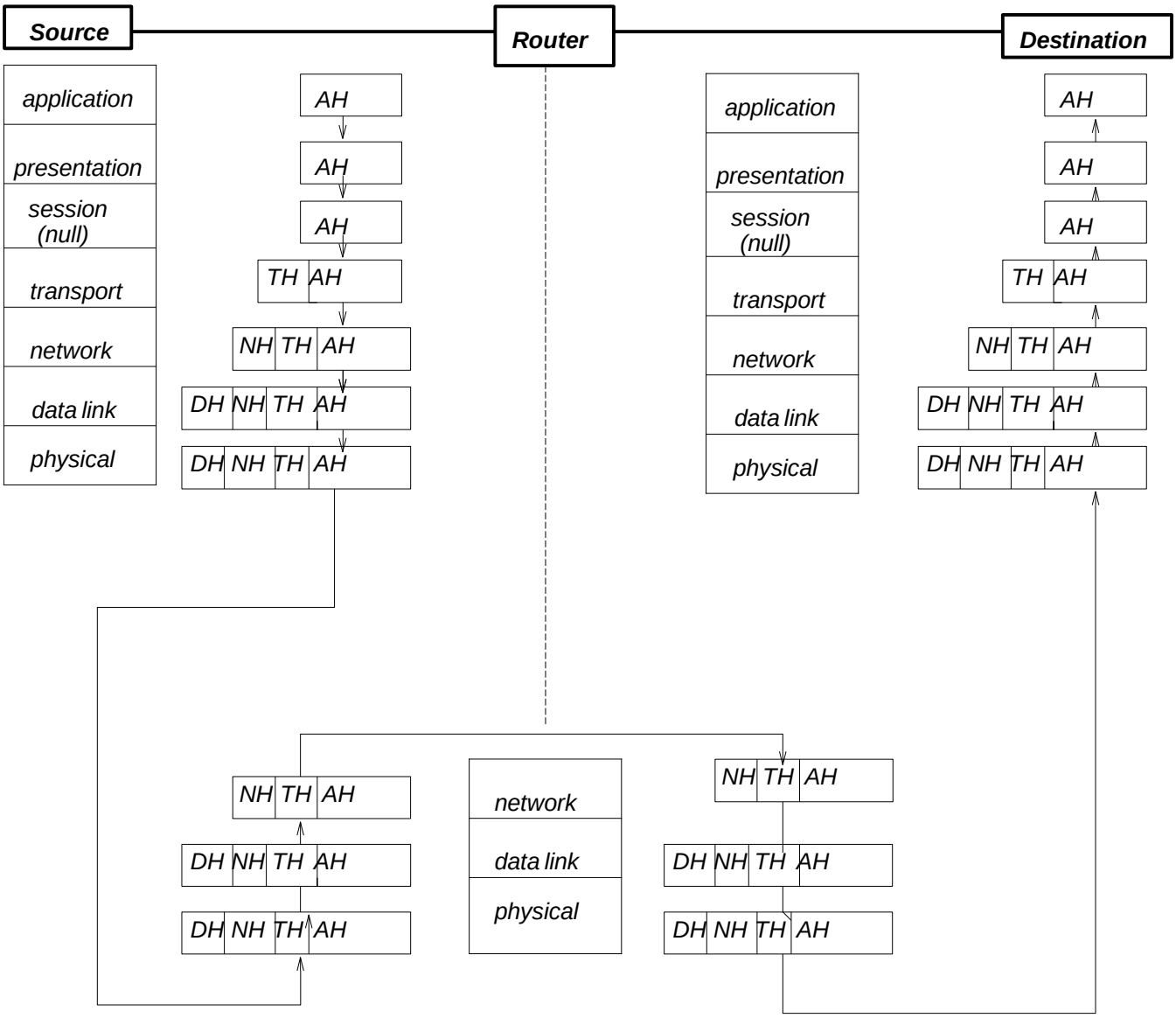
- Path MTU instead: Modern end-nodes find the right size (1500) instead of asking the routers to fragment. Fields ignored by most routers and in your project.

CS 118: IP Addressing and Overview and Project Intro

George Varghese



RECALL THE IP ABSTRACTIONS:

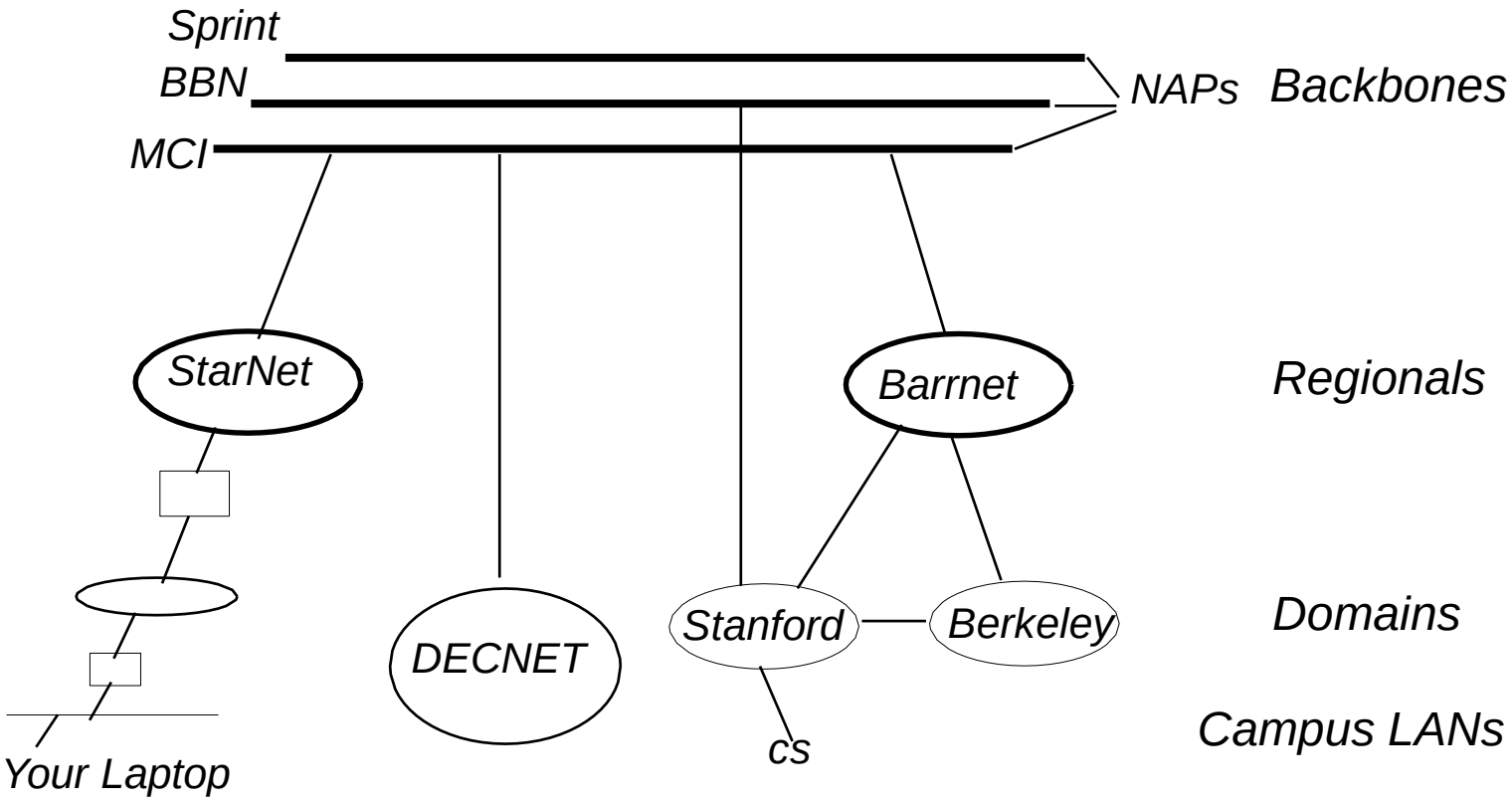


HEADERS: WATCH HOW THEY ARE ADDED AND REMOVED. STRICT LAYERING – THE DEFINITION

Life through the the Internet contains and endless set of unsorted pings. These prompts demand our attention, and present us with a perpetual onslaught of distractions and diversions . . . Our lives can easily be the story of how we respond to this endless set of prompts

From *Called*, by Mark Labberton, president of Fuller Seminary

TOPOLOGY



- Terminology: ISPs, POPs, Autonomous Systems, NAPs, Peering

Basic Internetworking in IP

- *Goal from start* Unlike DECNET, SNA etc. starts with a hierarchy of physical networks with network specific routing that IP does not care about, to create an Internetwork of physical networks
- *IP 's role* to route packets to the right physical network based on the network number. Offers a so-called datagram service with possible fragmentation and reassembly to deal with different maximum packet sizes
- *Error messages* companion protocol called ICMP for error messages (header checksum failed, maximum time exceeded, redirect etc.).

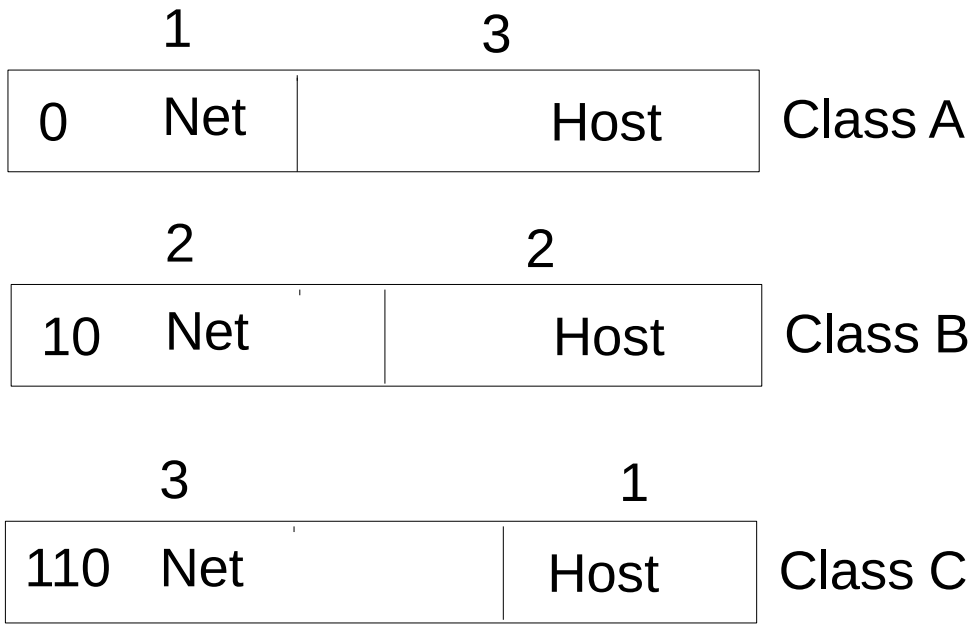
IP Evolution

- *ARPANET* started by linking government and university sites (including UCLA) in the 1970s
- *NSFNet* in 1983 ARPANET splits up into MILNET and ARPANET. In 1984 NSF establishes NSFNET to be backbone. Campuses attached to backbone via regional networks (NYSERNET etc.) Strict hierarchy breaks down because of direct connections between providers
- *Multiple Providers by lat 1980s* Internet becomes worldwide. From a research network to production quality. Multiple autonomous providers that need to work toi=gether

Names and Address

- *Names* when you send to a domain name like cs.Berkeley.edu , a resolver is your host translates the name to a 32-bit IP address. All messages carry IP destinations addresses
- *Domain Name Service* the translation is done using the so-called Domain Name Service or DNS which we will study later

ORIGINAL IP ADDRESSES: CLASSFUL



- Original Model: Small number of large networks (class A), moderate number of campus networks, large number of LANs
- Idea: Hierarchical address with a moverable boundary

Old IP Forwarding

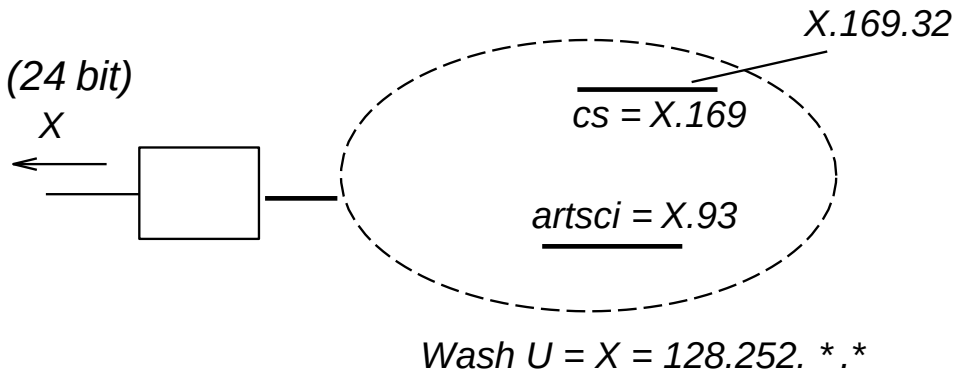
- *Find Destination* extract Network Number of destination address by parsing and checking for class A, class B etc,
- *Final hop reached?* If (Network Number of Dest = Network number of one of this router's local interfaces) deliver packet. Map to local address using ARP or some such network specific protocol
- *Lookup Router Table* Lookup Network Number in the corresponding routing table, If it exists, deliver packet to corresponding NextHop.
- *Lookup Router Table:* If no route entry exists, send to default router. (This looks silly but is a great way to avoid keeping lots of table entries in stub organizations like UCLA).

Challenge-Response

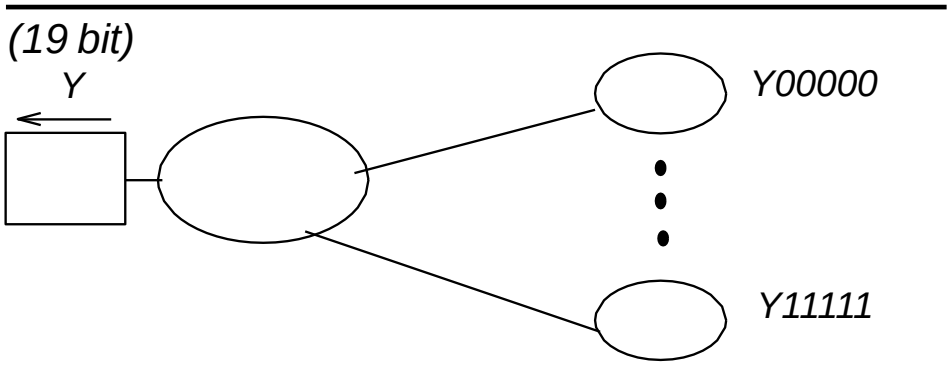
One level of hierarchy good but IP quickly ran into two scaling challenges:

- *Inefficient address usage:* any organization that needed more than 255 addresses asked for a class B address (64,000) and they quickly ran out
- *Routing Table Growth:* the response to no more class B addresses was to assign multiple Class C addresses. But now every backbone router needed to know more addresses, more routing traffic, search times etc.
- *Response* changed IP forwarding to longest matching prefix. Why?

SUBNETTING AND SUPERNETTING



Subnetting a Class B address X



Supernetting Class C addresses Y0−Y31

- Supernetting: Done recursively, leads to backbone routers only having hundreds of thousands of prefixes of lengths 8-32
- Temporary Measures: Often today new organizations are give 1 IP address and use NAT. Need the move to IPV6 (128 bits)

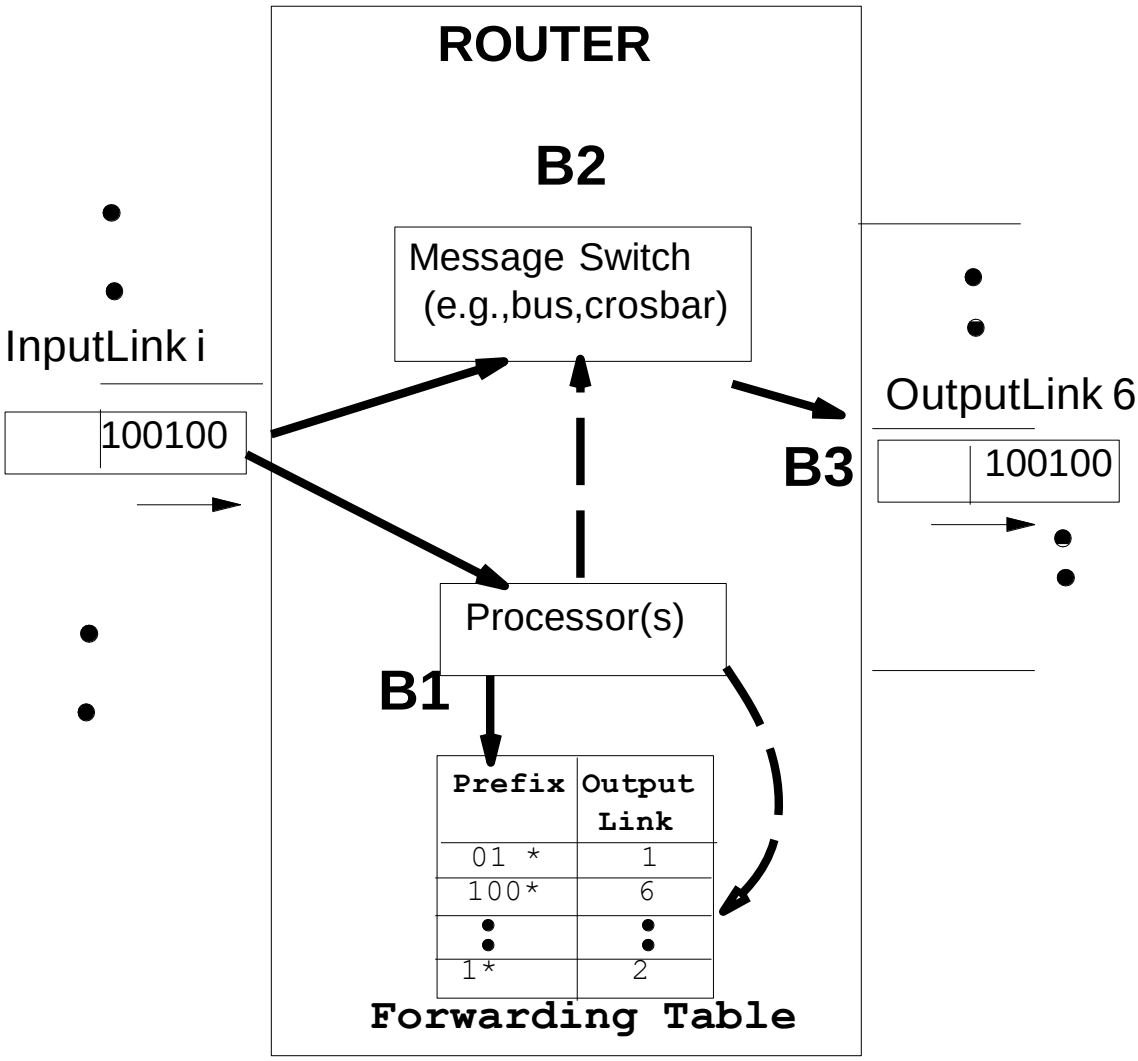
IP evolved to meet challenges

- *Challenge 1* Interconnecting diverse networks → Net Numbers (Class A, 1 byte)
- *Challenge 2:* Ethernets led to an explosion of networks → Hack to add Class B, Class C
- *Challenge 3* Class B addresses ran out → Give consecutive class C and use Longest Prefix Match
- *Challenge 4*: Even Class C's started running out → NAT and concurrent move to IP v6

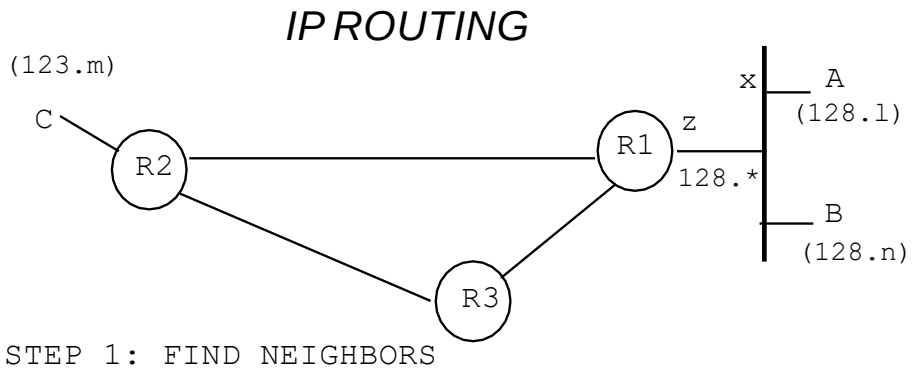
New IP Forwarding

- *Lookup* find longest matching prefix P of destination IP address in packet in forwarding table
- *Default or Local?* If P is nil forward on default route. If the next hop associated with P is a local interfaces, deliver packet. Map to local address using ARP or some such network specific protocol
- *Send on its way* if not, send packet to NextHop route associated with P
- Backbone routers in default-free zone have to have many hundred thousand prefixes to reach everyone. Enterprise routers have 1000s because of heavy use of default routes.

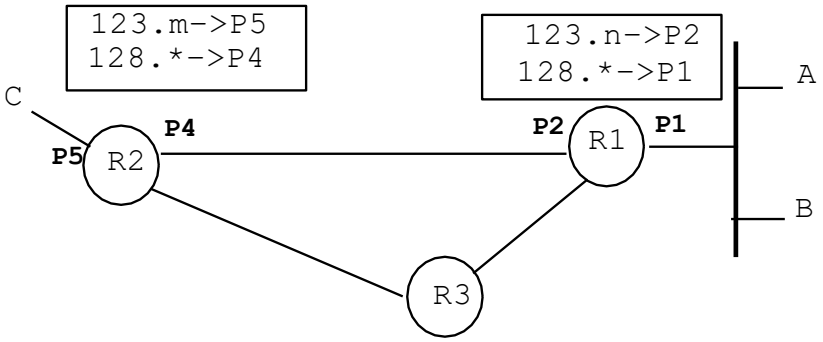
ROUTER MODEL



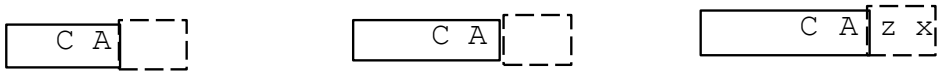
FORWARDING AND ROUTING



STEP 1: FIND NEIGHBORS

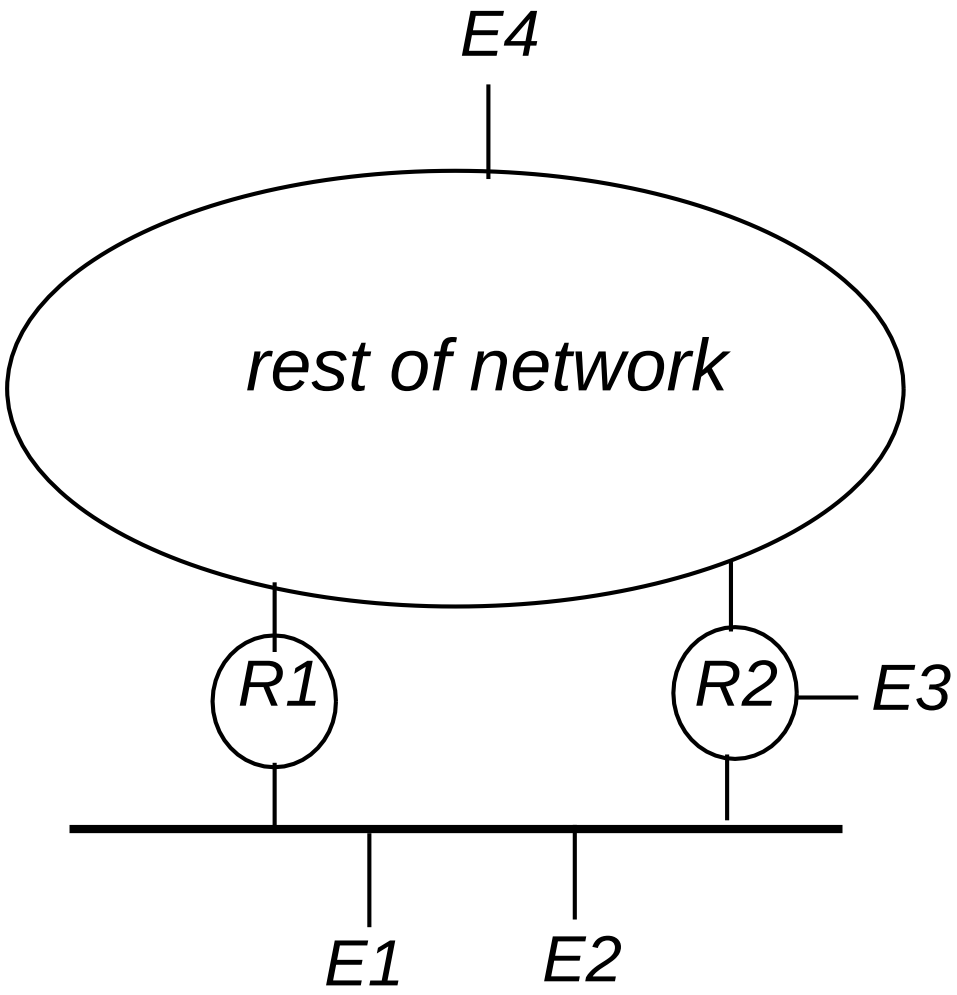


STEP 2: COMPUTE ROUTES



STEP 3: FORWARD

FOUR PROBLEMS ENDNODES MUST SOLVE



- P1: Routers need Data Link Addresses of endnodes
- P2: Endnodes need DL address of 1 router
- P3: E1 and E2 should be able to communicate without a router
- P4: E1 to E3 traffic should go through R2

IP Solutions to End-node Problems

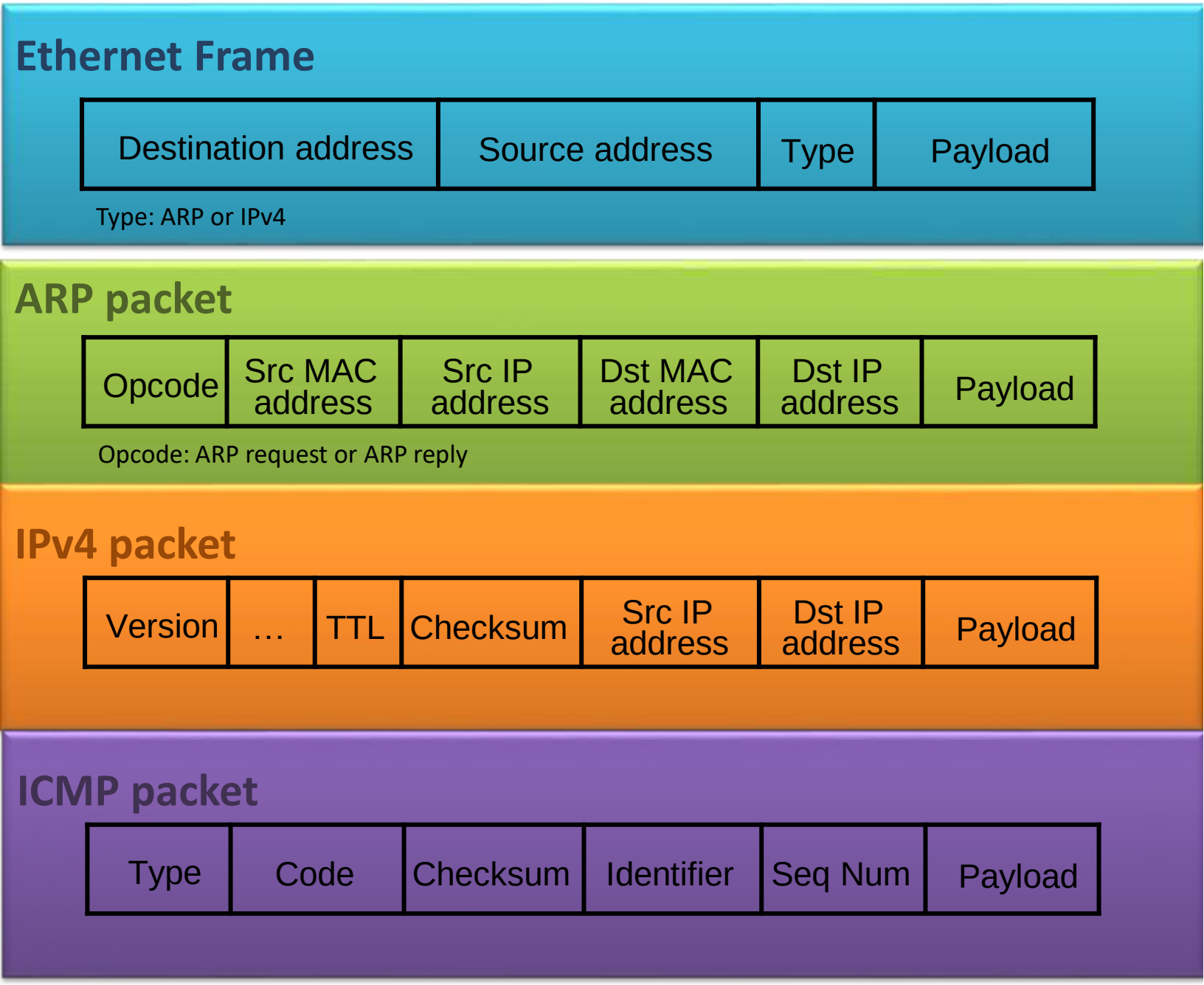
- *P1:* ARP for MAC address of destination
- *P2:* a service called called DHCP gives you the IP address of one router (autoconfiguration)
- *P3:* two endnodes know they are on same subnet by comparing masks. Then ARP
- *P4:* send to router and router sends redirect if packet returns on interface it entered router. (Ignore this code in project),

Transition to Project: IP Forwarding with ACLs

Overview

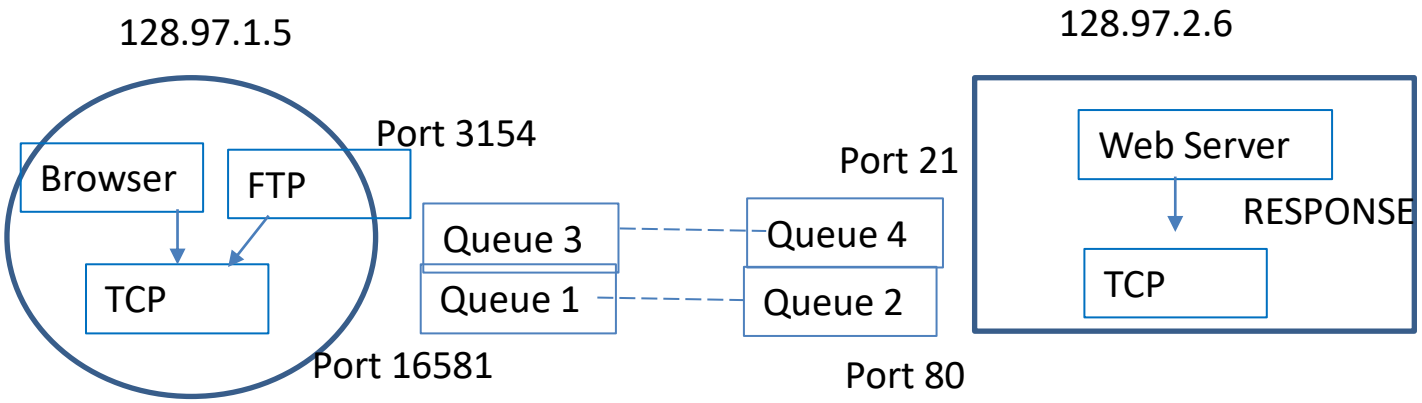
- *1:* Need to understand how to forward packets using longest matching prefix (linear search fine)
- *2:* Need to understand how to implement ARP in your router
- *3:* Need to understand how to implement ACLs using simple linear search
- *4:* To implement all this and ACLs, you need to understand all the fields in an IP Packet including the TCP header (though we will do TCP later)

Basic Packet Formats

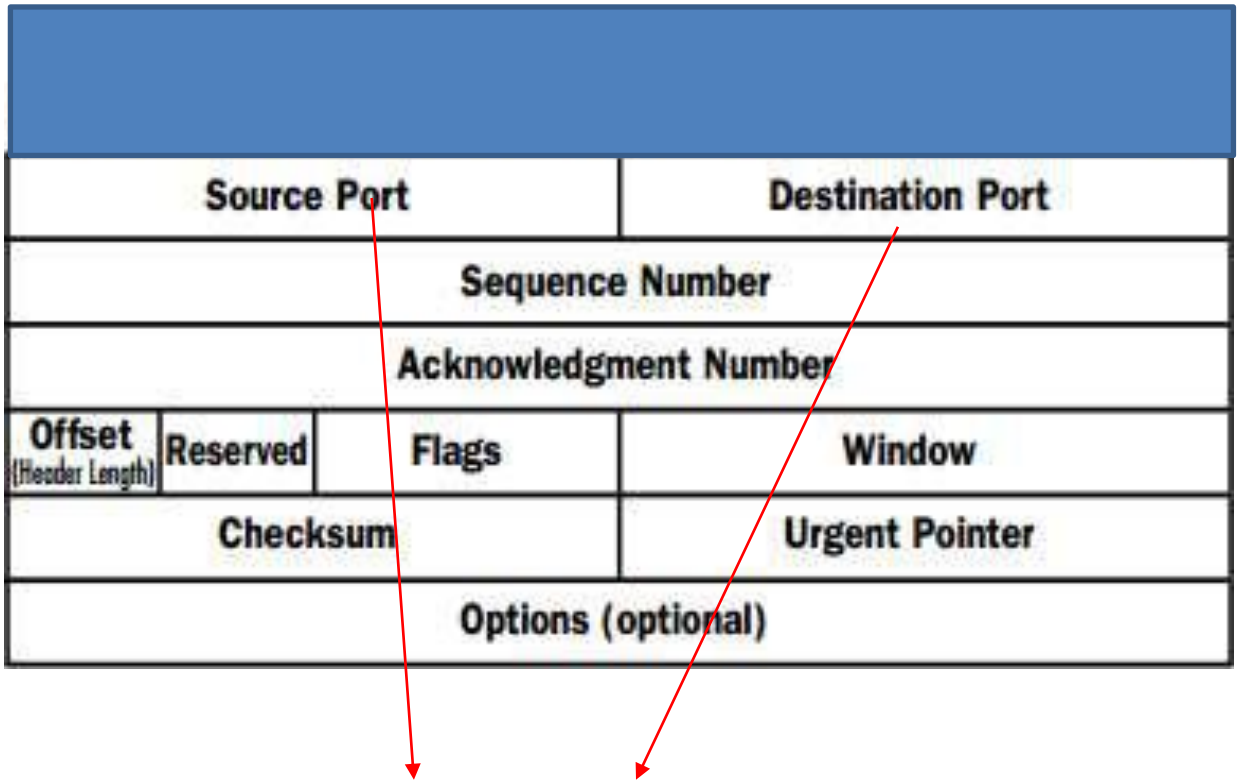


IPv4 header also contains header length, total length, ID, flags, fragment offset, and protocol fields. ICMP not Needed for this year's project

ACLs refer to TCP Ports:
Ports are like extensions



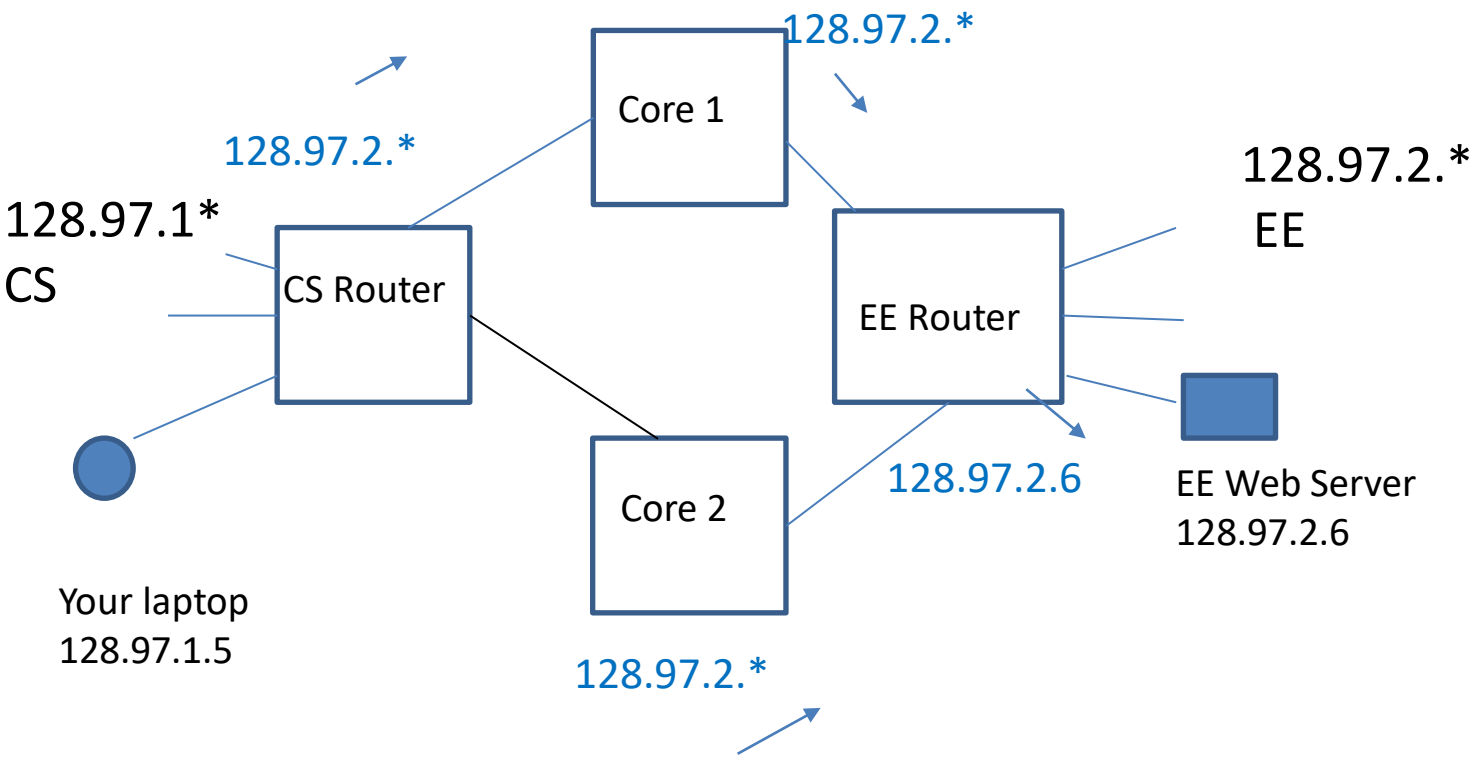
TCP Header: Where the Ports Lurk



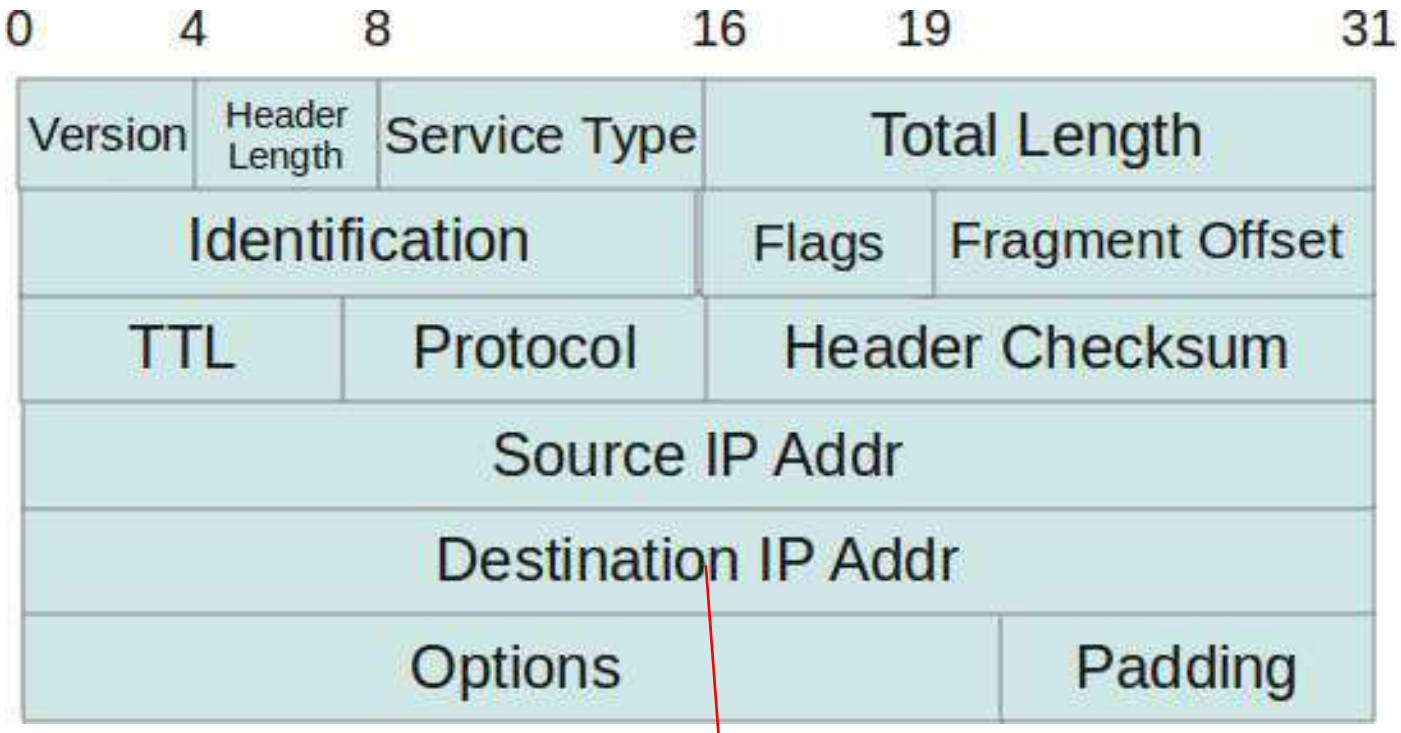
For ACLs these are the only TCP fields that matter as they identify the application and so routers/firewalls use these fields for forwarding. Other fields we will understand when we do TCP

Imaginary UCLA Topology

IP Forwarding (Data Plane)

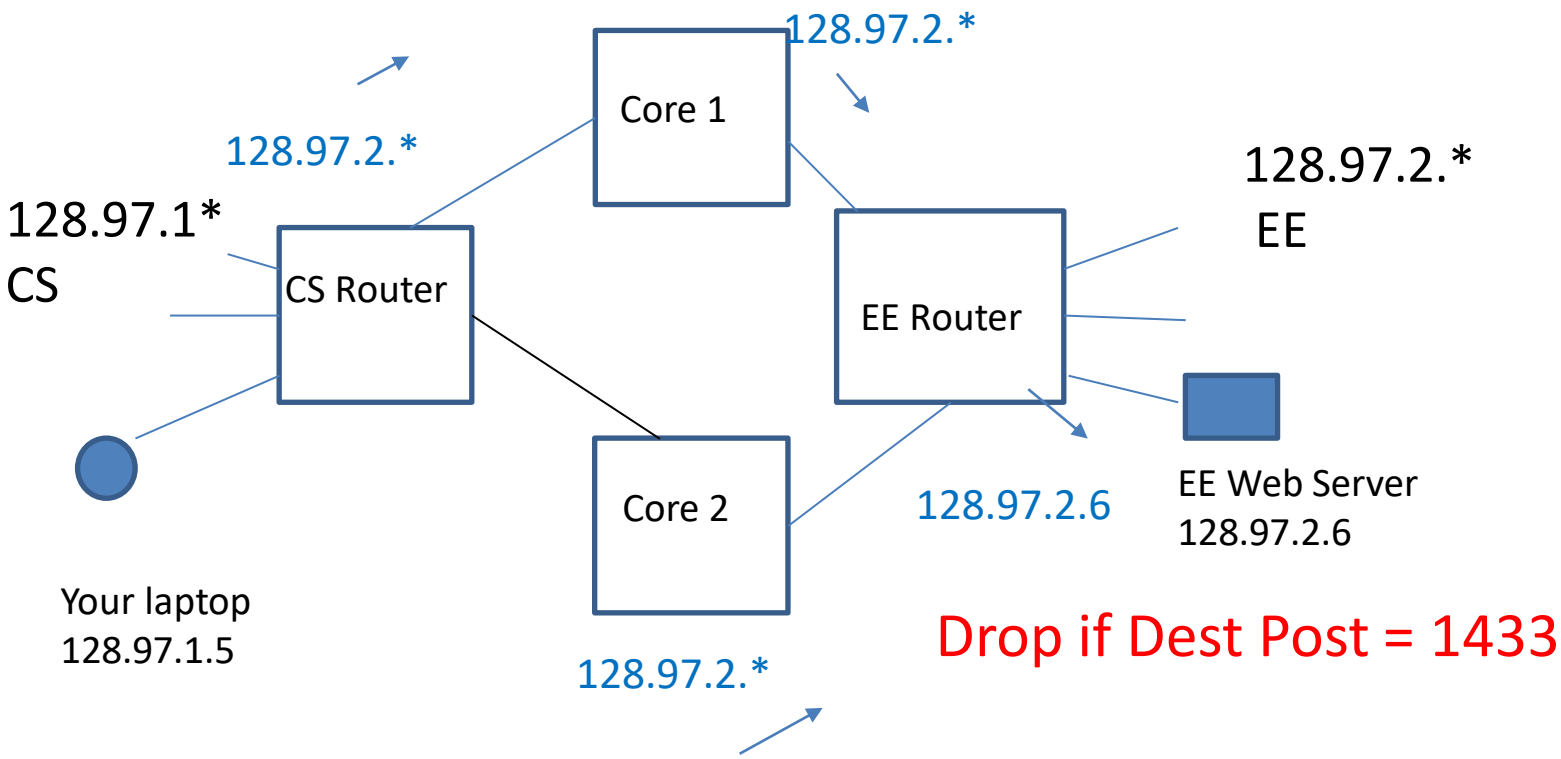


IP Header



We do longest match on the destination IP address: **main field**.
ACLs also check source IP address.
Also Protocol field and TTL

Controlling Forwarding with ACLS



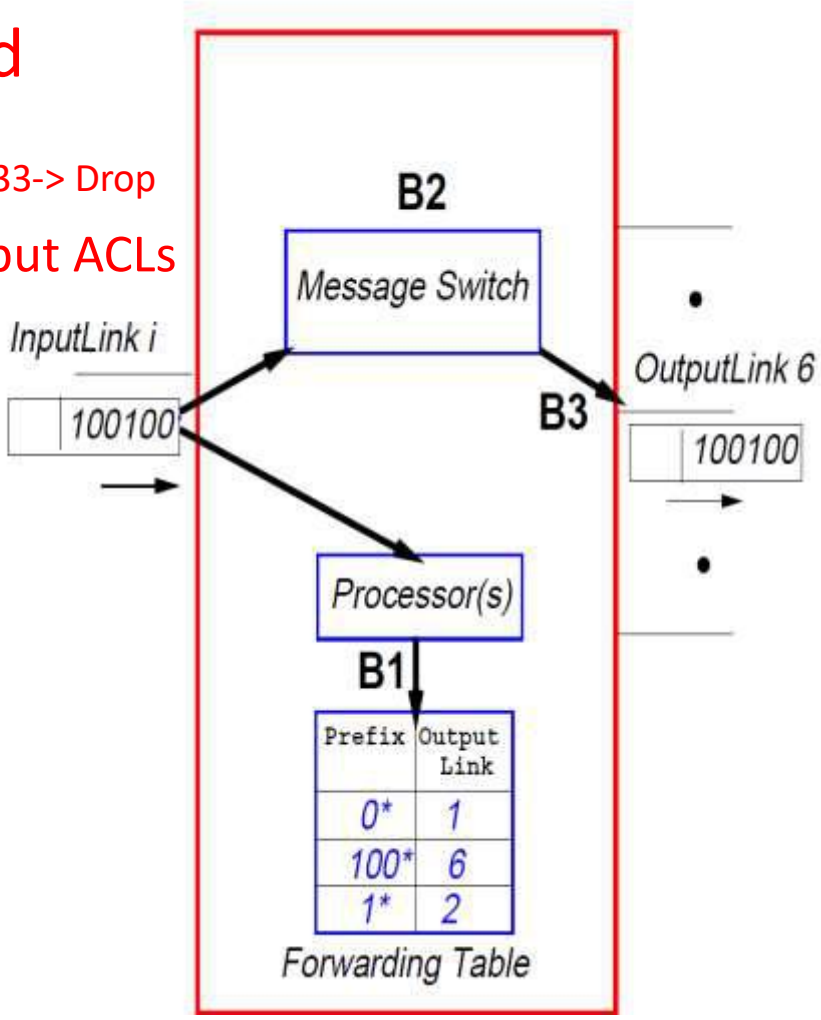
A famous attack called Slammer targeted the SQL Port 1433. After that it is common to block this port

Router Model with ACLs

Modified

Dest Port = 1433-> Drop

Input ACLs



ACL Syntax

Different for different vendors
Logically, a conjunction of predicates on IP and TCP fields and an action
For example, Dest IP = 129.97.* and Dest Port = 1433 → Drop
Most routers allow 100-1000 of them. When multiple match, first one wins (not longest as in Longest Prefix Match)

Forwarding Pseudocode: ARP

- 1. Find input network interface: findIfaceByName. Drop packet if interface is unknown

2. Read ethernet header and check the eth_type field. Ignore all but ARP and IPv4 types

3. If eth_type is ARP:

a. If ARP Request packet:

- Prepare and send ARP response packet

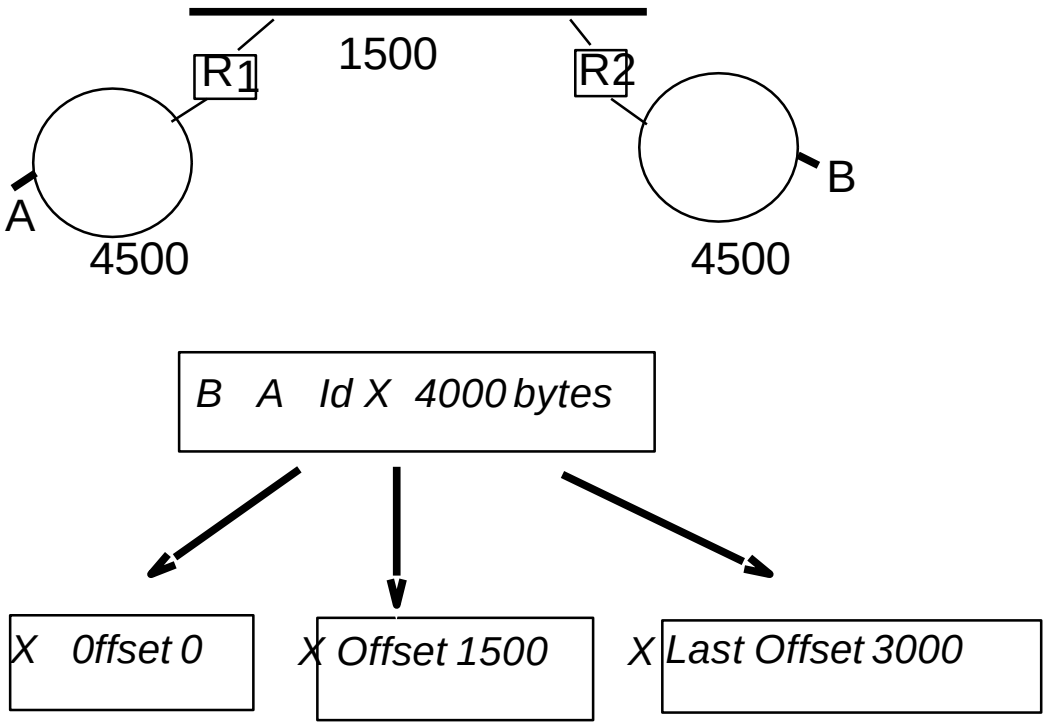
b. If ARP Response packet:

- record IP-MAC mapping information in ARP cache
- send out all enqueued packets for ARP entry

Forwarding Code: IPv4

- 4. If eth_type is IPv4:
  - verify checksum, length, discard invalid packets
  - Check 5-tuple (IP dest, IP source, protocol, dest, source ports) in input ACL and drop if needed
- 5. Use the Longest Prefix Match algorithm to find a next-hop IP address in the routing table
- 6. Lookup ARP cache for MAC address mapped to the next hop destination IP address
  -- If valid entry found: forward packet
  - Else: queue received packet and send ARP request to discover the IP-MAC mapping.

ASIDE ON FRAGMENTATION AND REASSEMBLY



- Path MTU instead: Modern end-nodes find the right size (1500) instead of asking the routers to fragment. Fields ignored by most routers and in your project.

Digression on Fast IP Lookups

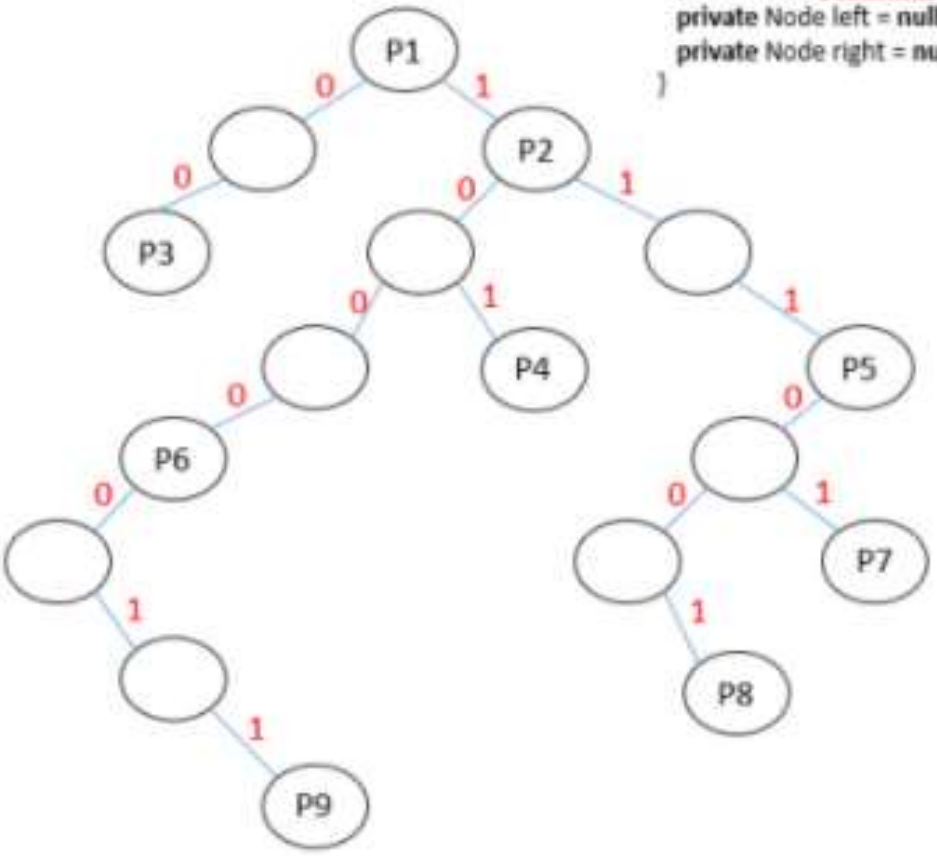
Approach 1: UNIBIT TRIE

https://raminaji.wordpress.com/unibit-tries/

Prefix database

P1	*
P2	1*
P3	00*
P4	101*
P5	111*
P6	1000*
P7	11101*
P8	111001*
P9	1000011*

```
private static class Node {
  private String nextHop = null;
  private Node left = null ;
  private Node right = null ;
}
```

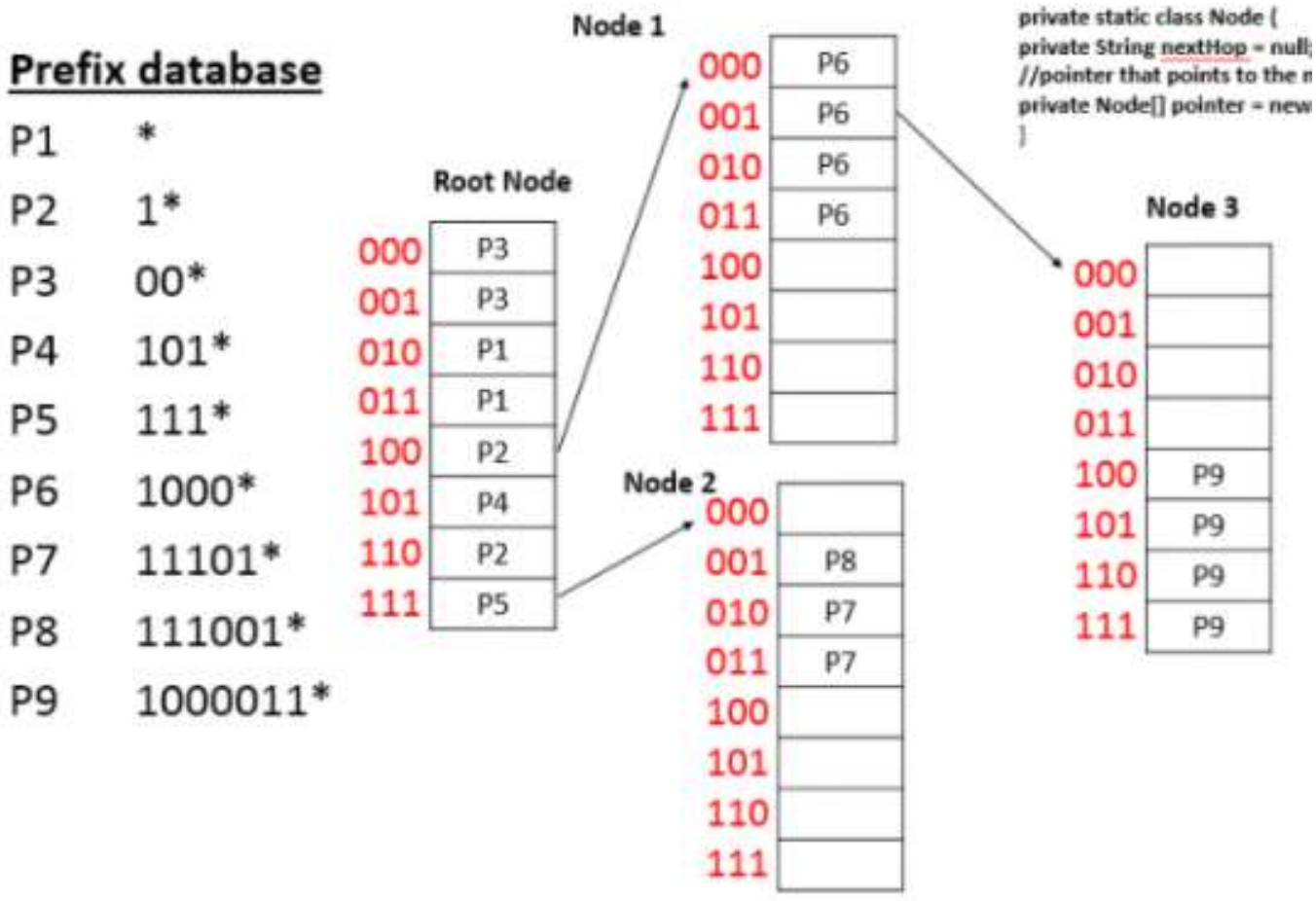


32 STEPS IN WORST CASE.

CONSIDERED TOO SLOW TODAY

2: MULTIIBIT TRIE

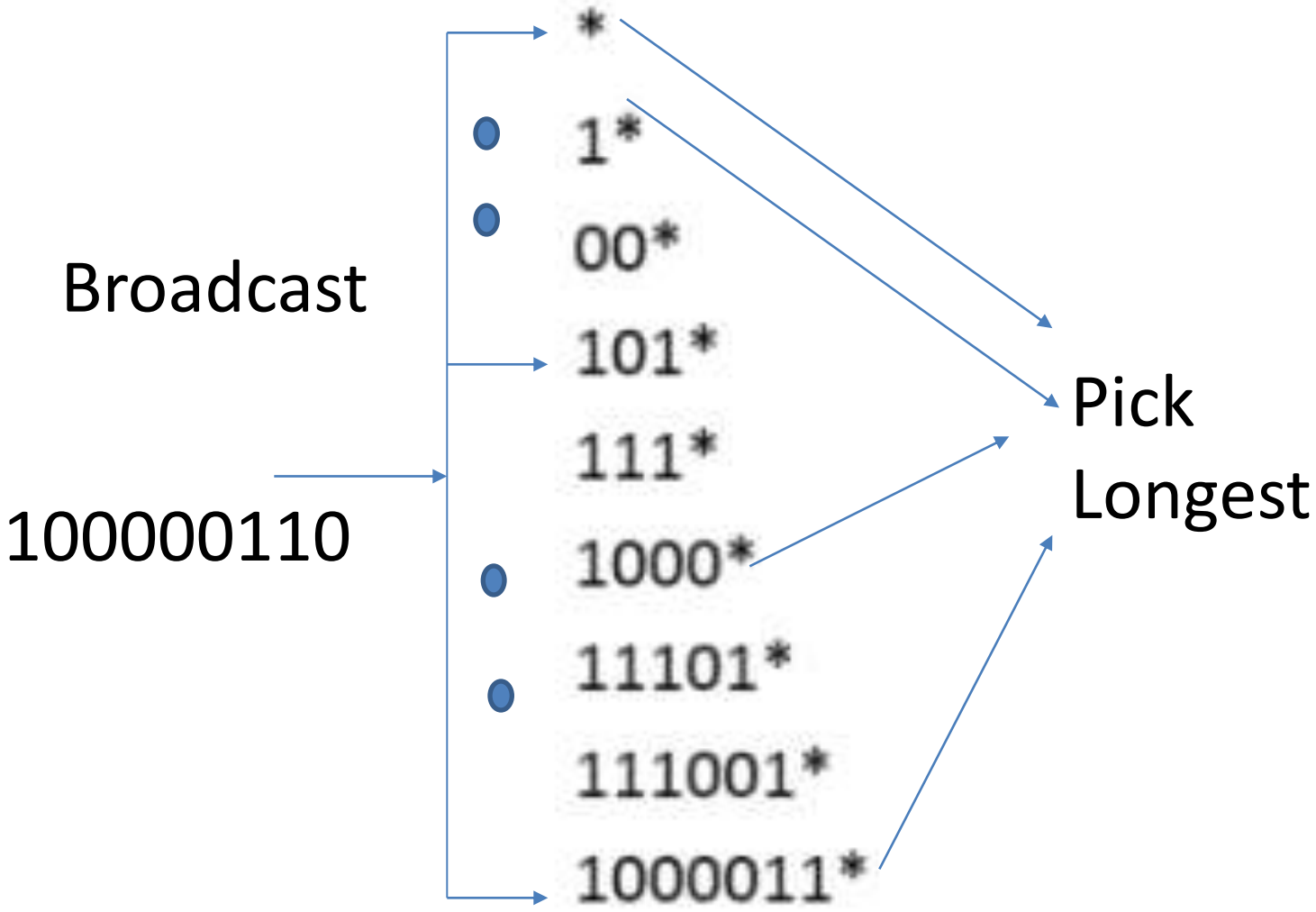
https://raminaji.wordpress.com/unibit-tries/ (Srinivasan-Varghese CACM 98)



11 STEPS IN WORST CASE. SLOW AND TOO MUCH MEMORY

3: TERNARY CAM

Memory where each bit can be 0, 1, * that can be search in parallel



1 STEP IN WORST CASE. BUT TOO MUCH POWER AT HIGH SPEEDS

B1: IP Lookup State of the Art

- *Compressed Multibit Tries* → Tree Bit map for Cisco CRS 1 at Terabits
- *CAMs for lower speed :* → Ternary CAMs (Barefoot)
- *More details* → Web site, Network Algorithmics text, my class