Week 9

TCP/IP stack

School of Information Technology and Electrical Engineering
The University of Queensland

Credits:

- Some slides from previous version
- Tanenbaum, “Computer Networks”
- Bryant and O’Halloran, “Computer Systems: A Programmer’s Perspective”

Protocols

Protocol – agreed rules of communication and behaviour.

- What can you say?
- How do you say it?
- What should happen when you do?

Layers

It often makes sense to build software on top of existing capabilities.

- Cleaner (although not necessarily more efficient)
- Lower layers can change without a complete rewrite.

Network Layers

Network code is written as a “stack” of layers. We will use the Internet (TCP/IP) stack as our example.

History:

- An internet is a connected group of different networks.
- Military origin

We'll show (some of) what is needed to implement http, ssh

Physical Layer

The medium through which signals travel.

(Data)Link Layer (2)

Talking to nodes which you can reach without an intermediary.

- Ethernet, WiFi, infrared, carrier pigeon
- Could be point-to-point or broadcast so we need addresses (MAC addresses).
- A node could have a number of link layer interfaces.
- Formats messages correctly and sends them over the physical layer.

Network Layer (3)

Communicating with any host on the "internet".

Two tasks:

- Find a node closer to the destination.
- Send the packet in that direction.
  - Delegated to the link layer.

The protocol for this layer is the Internet Protocol or *IP*. The address at this layer is the IP address.

Network Layer (3)

Why a different address

- Different link layers may have different rules for addresses. Need a new type of address that works everywhere.
- MAC addresses don't have hierarchy (that we can use).

So an Ethernet interface needs a MAC address and 1 (or more) IP addresses.

What's missing?

- All communication so far has been on the basis of packets. We may want streams.
- Can only talk to a computer not individual processes.
- Reliability? At the moment it's send the packet and hope.
  - Two generals?

Transport Layer (4)

Two main protocols (both use sockets and IP addresses):

- Transmission Control Protocol (TCP)
  - Connection oriented
  - Stream based.
  - Delivery is reliable:
    - In order.
    - Nothing missing
    - No duplicates.

UDP/IP

- User Datagram Protocol
  - Connectionless
  - Message based (think post card)
  - "unreliable"

If you don't mind messages going missing UDP is faster (no acknowledgement).

TCP will not move on until the current part of the stream has been delivered.

Application Layer (5)

Layer 5 is everything that builds on Layer 4:

- SSH, HTTP, bit torrent, skype, FTP
- Telnet (TCP), netcat(TCP and UDP)
- Your 4th assignment

TCP/IP Stack

5: Application Layer

4: Transport Layer

3: Network Layer

2: (Data)Link Layer

1: Physical layer

Reliability?

Checksums in Ethernet and IP headers mean that the receiver can can be "reasonably confident" that packets which arrive are not corrupt.

This is not the same as guaranteed delivery. [If you want that you need a higher layer eg TCP].

Headers and envelopes

Eg: A web browser(client) process wants a page from a server process.

- The client writes up an HTTP request to send to the server process.
- The server receives the HTTP request, and sends back the required page.

This is a communication (using HTTP) between two entities at level 5 [From the user's point of view].

Example continued

But layer 5 entities can't actually communicate directly. Instead they establish a socket connection using layer 4.

- The transport layer breaks the message into pieces and wraps them in a TCP header which has address and other information.
- On the other end layer 4 removes the headers, reassembles the message and hands it up to the HTTP code on layer 5.

So from the programmer's point of view this is a layer 4 communication simulating layer 5.

Example continued

But layer 4 entities rely on layer 3 to do the hard work.

- TCP messages are handed to layer 3 which adds IP headers and sent on.
- Headers from higher layers are not removed (layer 3 can't read them – they are just part of the message).

At this level we have two nodes communicating using IP.

...

Layer 3 doesn't actually send messages. It looks at the available link layer interfaces and sends the message out that interface using layer 2.

When layer 3 receives a message from layer 2, it either:

- keeps it, removes the IP header and passes it up to layer 4
- or updates the IP header and forwards it on.

....

Layer 2 adds a header (+footer for Ethernet) and sends it via the physical layer.

Note:

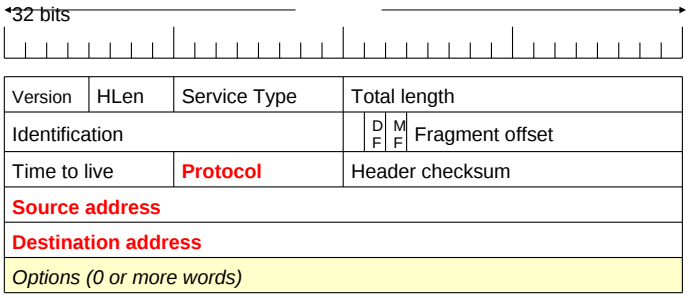
- You get more headers the at lower layers.
- Lower layers don't need to understand the headers of higher layers.
- We still talk about protocols at higher levels as if they were doing the work.
  - Because we don't care how they get it done.

Demo 1 and 2

IP Addresses

- 32-bit IP addresses are often written in **dotted-decimal** notation
  - each of 4 bytes written in decimal
    - e.g. 130.102.2.15
  - This notation used for human consumption
- Some addresses have special meanings
  - e.g., broadcast to all hosts on a particular network
  - More details later

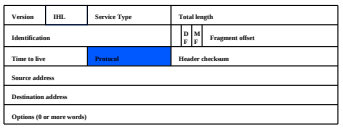
Internet Protocol (IP) Header



- We're interested in the **highlighted** header fields

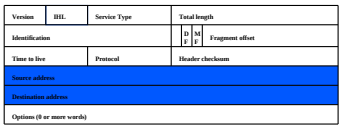
IP Header: Protocol

- 8 bits
- Range: 0 to 255
- Identifies higher level protocol to which packet should be passed, e.g.
  - 6 = TCP
  - 17 = UDP
- RFC 1700 – defines protocol numbers



IP Header: Source and Dest. Addresses

- 32 bits each
- Dest. address
  - Used to route packet to intended destination
- Source address
  - Destination can choose whether to receive
  - Destination knows whom to reply to



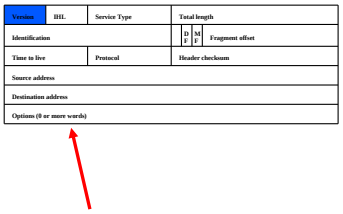
Internet Protocol (IP) Header

Version	HLen	Service Type	Total length
Identification		D F / M F	Fragment offset
Time to live	Protocol		Header checksum
Source address			
Destination address			
Options (0 or more words)			

- We'll look at a few more header fields today

IP Header: Version

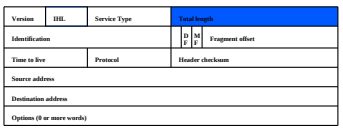
- 4 bits
- Specifies protocol version
- Allows protocol versions to change
- Currently v4
  - v6 around also (uses different header)



You're not expected to read these words. The figure just indicates which part of the header we're talking about (highlighted). (Refer to previous slide.)

IP Header: Total Length

- 16 bits
- Specifies total datagram length
  - includes header and data
- Maximum length = 2^{16}-1 = 65535 bytes
- BUT, in reality, datagrams are often limited by the underlying physical network, e.g. Ethernet

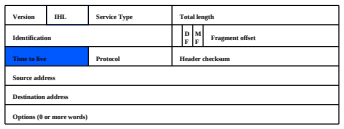


Question

- What's the largest amount of **data** that can be sent in an IP datagram?

IP Header: Time-to-live

- 8 bits
- Range: 0 to 255
- Limits packet lifetimes
- Decremented at each router
  - Was supposed to count seconds
  - In practice – counts hops
- Packet discarded when reaches zero



Time-to-live

- Why limit packet lifetimes?

Ping (and ICMP)

- demo in class.

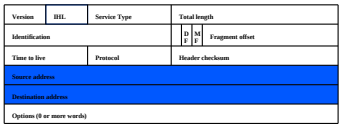
Aside – traceroute

- **traceroute** uses the ICMP protocol (built on IP) with varying Time-to-live values (starting from 1)
- The expiry messages returned allow the route through the network to be determined

Break

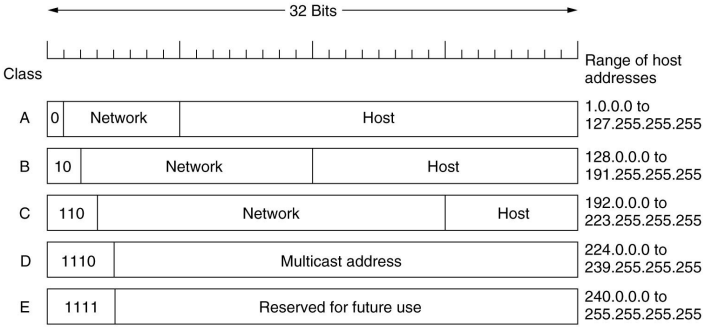
IP Header: Source and Dest. Addresses

- 32 bits each
- Dest. address
  - Used to route packet to intended destination
- Source address
  - Destination can choose whether to receive
  - Destination knows whom to reply to



IP Addresses – classful addressing

- Each host (network interface) has a unique 32 bit address
- Hierarchical format: (network number plus host number)



Why?

- Why are IP addresses hierarchical?
- Why don't we just use Ethernet (MAC) addresses?

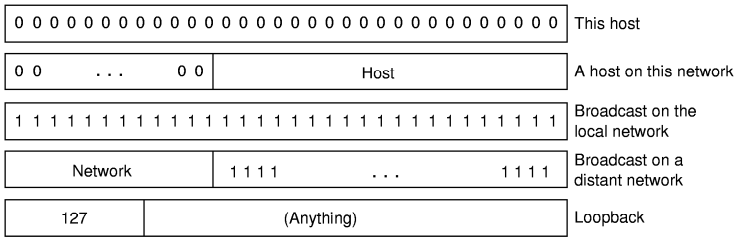
IP Address Classes

- **A** – up to 126 (2^7-2) networks
  - 16 million (2^{24}-2) hosts each
- **B** – up to 16382 (2^{14}-2) networks
  - 64k (2^{16}-2) hosts each
- **C** – up to 2 million (2^{21}-2) networks
  - 254 hosts each

Dotted Notation

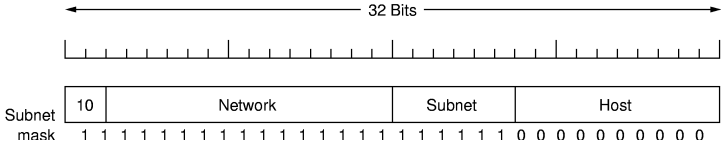
- IP addresses written in **dotted-decimal** notation
  - each of 4 bytes written in decimal
    - e.g. 130.102.2.15
- Addresses with all 0's and all 1's have special meanings
  - 0 = this network or host
  - 1 = broadcast – all hosts on indicated network

Special IP addresses



Subnets

- **Subnet** – split a network into parts
- Still appears as single network to outside world
- Example – class B network:



- Doesn't have to happen on 8-bit boundary
- This example – 6 bit subnet ID, 10 bit host ID - allows 62 LANs, each with up to 1022 hosts

Forwarding / Routing

- Routers have (incomplete) listings of
  - network-num.0
  - this-network-num.host-num

  addresses
- Each entry associated with network interface to send packet out on
  - Process is called **forwarding**
  - (Note: Each router interface has different IP address)
  - (**Routing** is the process of building up the tables of information)
- If network not listed – packet sent to some default router
- Routers only need to know about
  - local hosts
  - some other networks

Forwarding and Subnets

- Forwarding table entries – slightly different form
  - this-network-num.subnet-num.0
  - this-network-num.this-subnet-num.host
- Router on subnet k knows how to
  - send packets to other subnets
  - send packets to hosts on subnet k
- Subnet k router *doesn't* need to know about hosts on other subnets
- Forwarding table sometimes called routing table

Subnet masks

- Mask that removes host-id when ANDed with address
  - Bitwise AND
- Example address: 130.50.15.6
  Subnet mask: 255.255.252.0
- Destination subnet: ___.___.___.___
- Subnet address is looked up in forwarding table

Non-routable IPs

RFC 1918– Address Allocation for Private Internets

```
10.0.0.0    …  10.255.255.255
172.16.0.0  …  172.31.255.255
192.168.0.0 …  192.168.255.255
```


UQ Network

- Class B
  - 130.102.*.*
- Most subnets used to be at 8 bit boundary
- Now have been rationalized
- Old ITEE (then CSEE) subnets
  - 130.102.16
  - 130.102.48
  - 130.102.64
  - 130.102.65
  - 130.102.96
  - 130.102.180
  - 130.102.192
  - 130.102.193
- Current ITEE subnets
  - 130.102.64-67
    - (10 bit host number)
    - Staff/postgrad network
  - 130.102.72-75
    - (10 bit host number)
    - Student network
  - 130.102.79
    - (8 bit host number)
- Main UQ routers don't need to know all hosts on campus
- Just need to know about subnets

Hosts and Forwarding

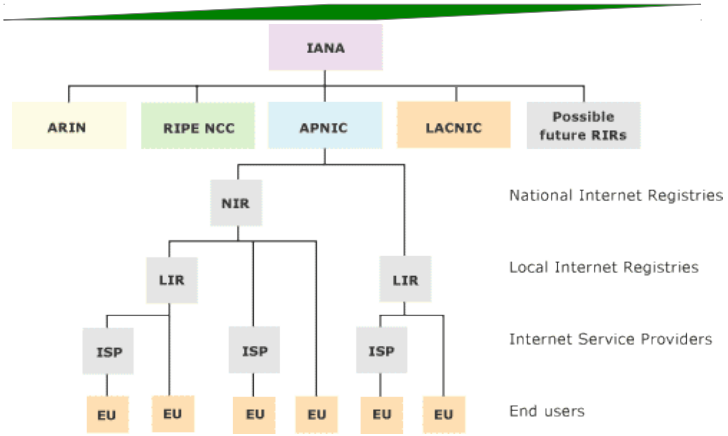
- Hosts with multiple network interfaces do forwarding too
- PC example
  - `route print`

CIDR – Classless Inter-Domain Routing

- A,B,C classful networks too inflexible
- IP address blocks now allocated in a classless manner using a hierarchy of registries
- Networks expressed as
  base IP address/N
  where N is the number of bits identifying the network part of the address
- Examples
  - 58.0.0.0/15 belongs to Fujitsu
  - 125.128.0.0/11 belongs to Korea Telecom
  - 203.48.0.0/14 belongs to Telstra

[From www.apnic.net]

Registry Hierarchy



[From www.apnic.net]

CIDR and Routing

- ISPs can allocate blocks of addresses within the blocks that have been allocated to them
- Routers outside the ISP only need to know about the common prefix
- This is called **routing prefix aggregation**, or **supernetting** or **route summarization**

NAT - Network Address Translation

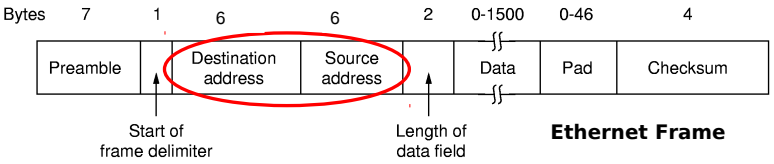
- One approach to deal with shortage of IP addresses
- Basic idea
  - Assign an entity (organisation) a single IP address
  - Use unique, private IP addresses within organisation
    - These same addresses can be used within multiple organisations
  - Change private IP address into organisation's IP address when packet leaves network
- Three ranges of private IP addresses exist
  - Details in class
- Private IP addresses must not appear on Internet

NAT - Network Address Translation

- One approach to deal with shortage of IP addresses
- Basic idea
  - Assign an entity (organisation) a single IP address
  - Use unique, private IP addresses within organisation
    - These same addresses can be used within multiple organisations
  - Change private IP address into organisation's IP address when packet leaves network
- Three ranges of private IP addresses exist
  - Details in class
- Private IP addresses must not appear on Internet

Addresses on the Network

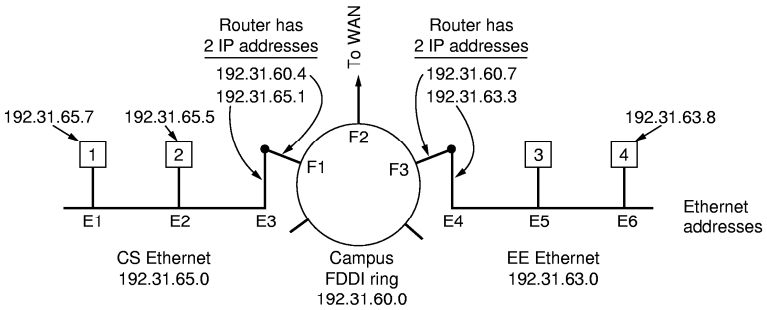
- Ethernet doesn't understand IP addresses
- Actually need to send info with MAC address
- How do we map IP addresses to LAN addresses?
  - Static – have a configuration file or table
  - Dynamic – ask over the network



ARP: Address Resolution Protocol

- Example: Host needs to send to 192.31.65.5 (already known to be on local network)
  - Sends broadcast packet:
    - "Who owns IP address 192.31.65.5?"
  - ONE reply should come back
- Variations
  - Cache
    - Entries should expire every few minutes
  - Supply own details when make request
  - Broadcast on boot
    - "Who owns my IP address?"

Example



- Several class C networks (how do we know this?)
- How does an IP packet get from
  - host 1 to 2? host 1 to 4?