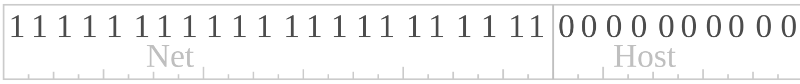
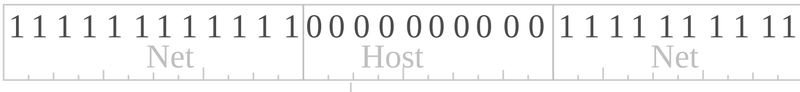
# Subnet Masks

- RFCs 917 922 925 (1984) 932 936 940 950 (1985)
  - First major change to IP after RFC791


- Subnet Mask




  - Bits set indicate net number
  - Bits clear indicate host
- Non-contiguous masks




  - Was used & Works fine
  - No advantages, Harder to understand
    - Abandoned

# Address Boundaries

- To Administrator

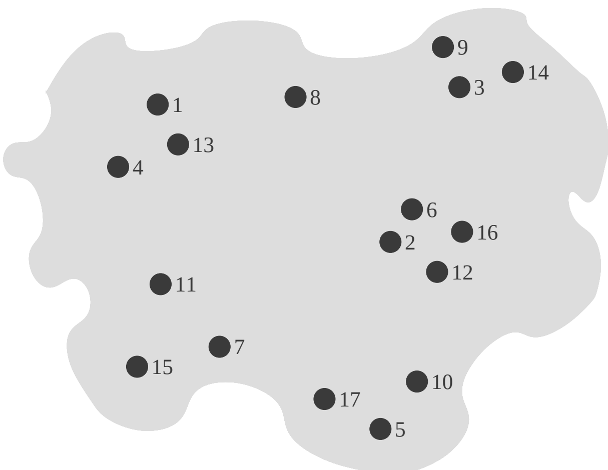

- Outside the network


- Inside the Network


- Implementation sees just one boundary
  - net / host
    - except at border

# Address Assignment

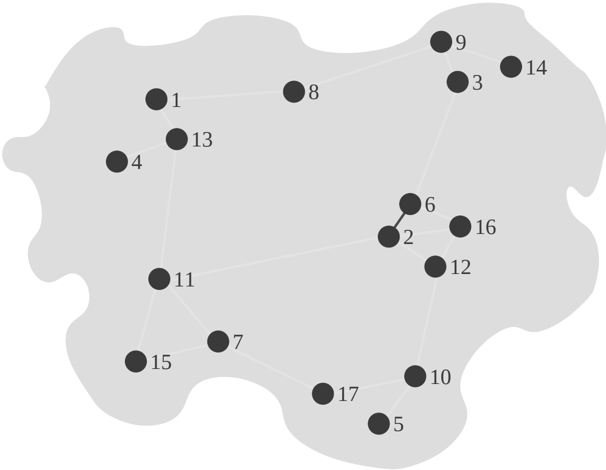
- As network grew...




- Addresses assigned arbitrarily
  - in allocation order
    - now meaningless

# Impact of Assigment Method

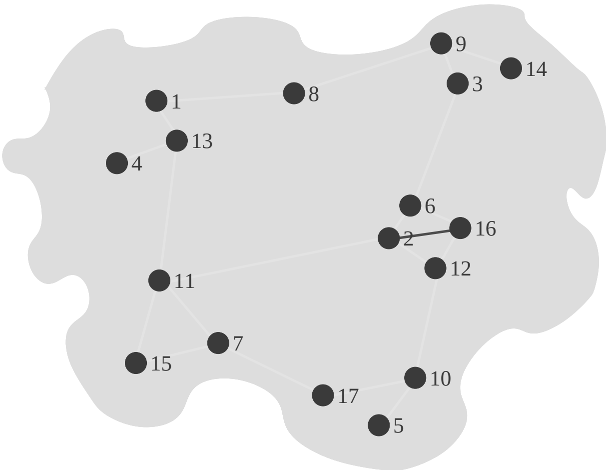
- Addresses used
  - To identify nodes
  - To find path to reach nodes
- Eg: From node 2:




  - To reach node 6
    - send directly on link to 6

# Impact of Assigment Method

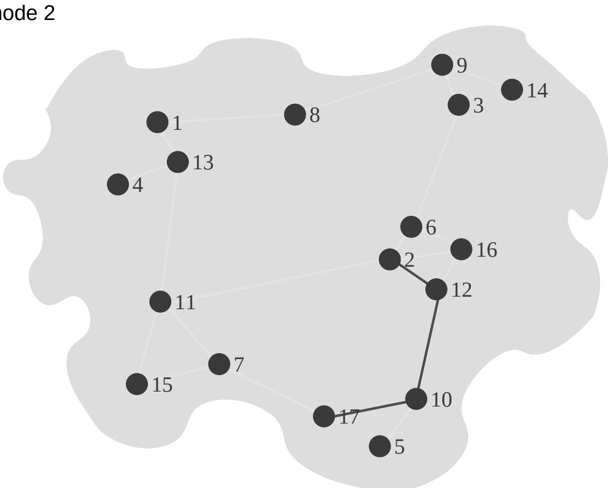
- Similarly (from node 2)




  - To reach node
    - 11: send directly to 11
    - 12: send directly to 12
    - 16: send directly to 16

# Impact of Assigment Method

- And for other destinations
  - From node 2




  - To reach node
    - 1: send to node 11 (it will forward)
    - 3: send to node 6
    - 4: send to node 11
    - 5: send to node 12

# Routing impact

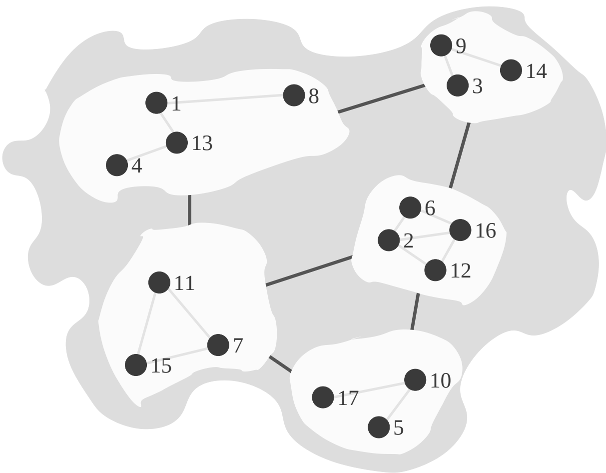
- Node 2:

| to | via | | to | via |
|---|---|---|---|---|
| 1 | 11 | | 3 | 6 |
| 4 | 11 | | 5 | 12 |
| 6 | 6 | | 7 | 11 |
| 8 | 6 | | 9 | 6 |
| 10 | 12 | | 11 | 11 |
| 12 | 12 | | 13 | 11 |
| 14 | 6 | | 15 | 11 |
| 16 | 16 | | 17 | 12 |

- Each node needs
  - list of all other nodes
  - and next hop to use
    - to reach the other node

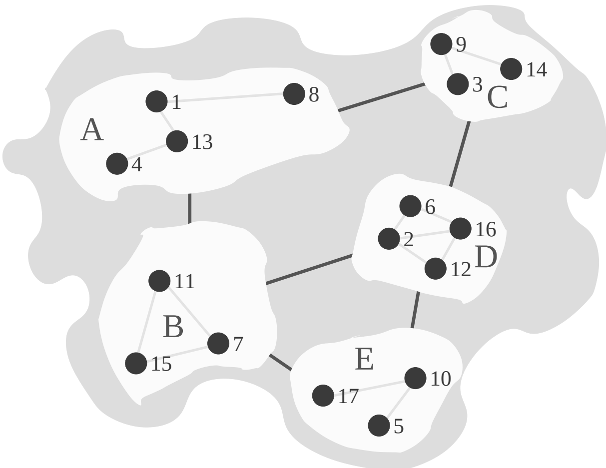
# Net Development

- Service Providers appeared




  - Simplify nodes routing
    - Know routes to neighbours
    - Everything else
      - Send to ISP

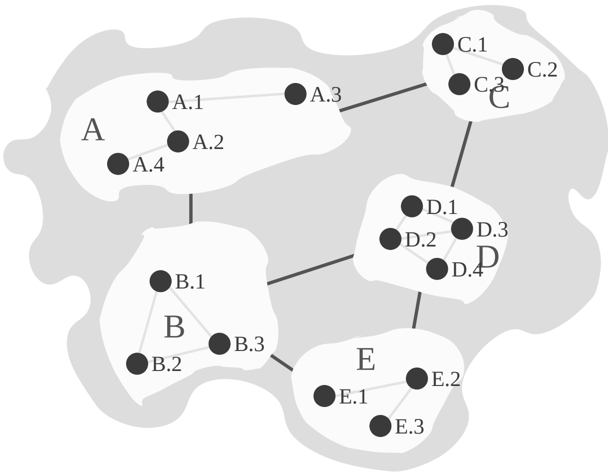
# Service Provider Routing




- Service Provider
  - most know location of
    - every network node
    - which ISP to send to

# Better Method

- Reassign all addresses




- Now
  - Address makes it clear
    - which ISP it belongs to

Hierarchical Addressing

# IPv6 - Why?

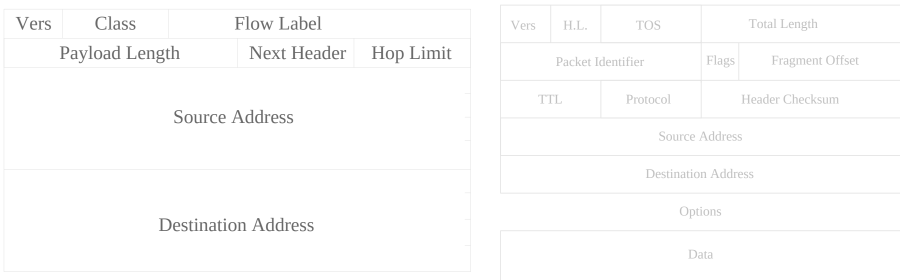
- IPv4 running out of addresses
  - 2^32 ==> 4 Billion addresses max
  - But many "wasted" because of allocation policy
  - 2 million connected organisations max
- IPv4 routing tables becoming unmanageable
  - ~50K routes
    - now ~200K routes
  - needs recalculating frequently

# IPv6 - Answers

- IPv6 addresses 128 bits
  - 2^128 ==> too big for words to describe
  - 340282366920938463463374607431768211456
  - (39 digits)
    - Very Very many will be wasted
- Routing
  - Later...

# IPv6 Header

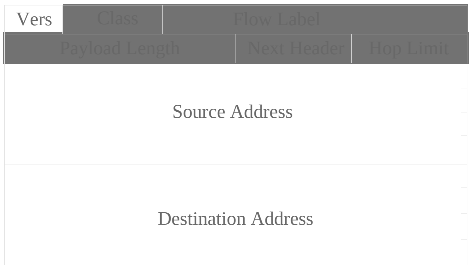

# Header Comparison

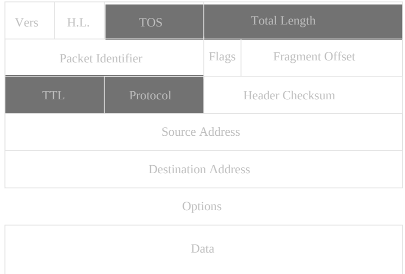



# Equivalent fields


- Version
  - 4 for IPv4
  - 6 for IPv6
    - Hence the names...
- Version identification is one way to allow upgrades

# Modified Fields

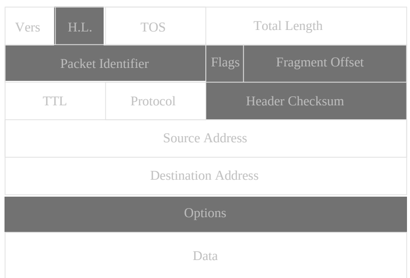



# The Modified Fields

- Time to Live
  - Hop Limit
- Total Length
  - Payload length
- Protocol
  - Next Header
- TOS
  - Traffic Class (DSCP) & Flow Label
    - Currently not well defined

# Missing Fields

# The Missing Fields

- Header Length
  - IPv6 header is fixed length ==> not needed
- Options
  - Exist in a different form
- Fragmentation fields (ID / Offset / Flags)
  - Exist in a different form
- Header Checksum
  - The controversial change - gone completely

# An IPv4 Packet

```
0:10:a4:f:41:cf 0:1:3:40:8a:e5 0800 98:

    172.30.0.77 > 172.30.0.161:
                 icmp: echo request

      4500 0054 7e24 0000 ff01 e459 ac1e 004d
      ac1e 00a1 0800 a334 0d82 0000 b613 3a3b
      5ff7 0c00 0809 0a0b 0c0d 0e0f 1011 1213
      1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
      2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
      3435 3637
```

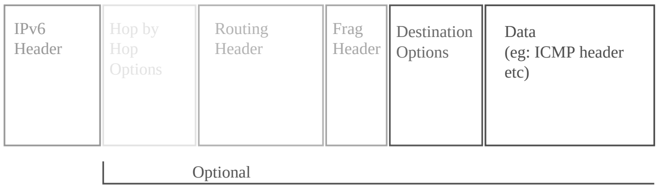
# An IPv6 Packet

```
0:10:a4:f:41:cf 0:1:3:40:8a:e5 86dd 70:

    3ffe:8001:2:181:210:a4ff:fe0f:41cf >
     3ffe:8001:2:181:201:3ff:fe40:8ae5:
                  icmp6: echo request

      6000 0000 0010 3a40 3ffe 8001 0002 0181
      0210 a4ff fe0f 41cf 3ffe 8001 0002 0181
      0201 03ff fe40 8ae5 8000 e8a3 0d9c 0000
      7616 3a3b d908 0700
```

# IPv6 Header Chains




- Each Header (before data) contains a "Next Header" field
  - Specifies the type of the header that follows
  - The first header is always an IPv6 header
- Each Header is of fixed length, or provides a mechanism to allow its length to be determined
  - Length field in most headers

# Header Processing

- Headers are processed "left to right" through the packet
  - Unrecognised header is an error,
    - ICMP error report
    - Drop packet
- Routers look only at IPv6 header, and Hop-by-Hop Options
  - Easy to tell if HBH is present
    - Next Header field in IPv6 Header (0)

# Options Header




- Header Length
  - Counts number of 64 bit words after the first
  - All (new) IPv6 headers are a multiple of 8 bytes
- Each option contains Type Length (and Data)
  - Can skip unknown options

# IP Options

- Some options for destination node
- Some options for routers along path
- Some options for some routers along path

- IPv4 options
  - In IPv4 header
  - All routers must examine all options
    - Only way to find options that need processing

- Many IPv4 options in RFC791
  - Source Routing
  - Route record
  - Timestamp record
- Almost none in use now

# IPv6 Options

- Multiple Option headers possible

- Options go in header that will be seen by appropriate node


  - All routers look at options in "Hop By Hop"
    - Special case
  - When Dest Addr in IPv6 header is reached
    - Routing options processed
    - Routing header processed
    - New IPv6 dest addr inserted
  - When routing header runs out
    - Final destination
    - Destination options processed

# Using options

- Using IP header option in IPv4
  - Causes every router to have to examine options
  - Causes packet to not be hardware forwarded
  - Slows processing - sometimes changes routing
    - 2nd class packet
  - Hence options not used

- 40 byte max option space
  - Not enough for interesting source routes
  - not enough to record routes, or timestamps
  - hence options not used

- IPv6 suffers neither of those problems

# IPv6 Addresses

- 128 bits
- Like IPv4, divided into
  - network identifier
  - host identification on that network

  Eg:
  172.30.2.60 netmask 255.255.255.0

  172.30.2.0 is the network identifier
  0.0.0.60 is the host identifier on that network
- Netmask can divide anywhere, not just at one of the dots

# IPv6 Prefix

- The "network part" of an IPv6 is the prefix
- Prefix identified by a length, rather than a mask
  - written /nn (as in /32 or /57)
- Prefix is never longer than 64
  - On a standard network
- That is, 64 bits are always available to identify a host on the network

# IPv4 Host Configuration

- Host must be configured with
  - IP (IPv4) Address
  - Netmask
  - Address of a router to use
  - (other info usually, eg: DNS server)
- Manual configuration
  - Difficult
  - Error Prone
- DHCP
  - Dynamic Host Configuration Protocol
  - Successor to BOOTP

# DHCP

- Will examine DHCP in more detail later
- However ...
  - DHCP requires a server
    - That server must be configured
      - OK for large nets (university, company)
      - Not great for small nets (home, shop)
  - DHCP server
    - can implement any address assignment policy desired
      - SMOP ("simple matter of programming")
  - eg: assign addresses only to "known" hosts
  - or assign addresses to anyone who asks

# IPv6 Autoconfiguration

- The 64 bit host part is big enough to contain a MAC address
  - Not an accident
- Host is shipped with MAC address available from hardware
  - PROM or similar
- If host can learn the prefix of the local network, it can create its own IPv6 address by simply putting its MAC address in the low 64 bits
  - MAC address is unique (enough), so the address formed should be unique too.

# IPv6 Packet Example

```
0:10:a4:f:41:cf 0:1:3:40:8a:e5 86dd

 3ffe:8001:2:181:210:a4ff:fe0f:41cf >
  3ffe:8001:2:181:201:3ff:fe40:8ae5
      icmp6: echo request
```

- Note similarity between MAC and IPv6 addresses
  - Both source and destination

```
                    00:10:a4      :0f:41:cf
   3ffe:8001:2:181:02 10:a4 ff:fe 0f:41 cf
```

# Generating IPv6 Addresses

```
00:10:a4      :0f:41:cf
02 10:a4 ff:fe 0f:41 cf
```

- FFFE is the defined way to transform a 48 bit MAC address into a 64 bit MAC address (EUI-64)
  - FFFE is inserted between the 3rd and 4th octets of 48 bit address
- 02 in the first address octet is the "locally defined" bit
  - In MAC address bit set indicates a locally defined address
  - In IPv6 address, bit is inverted

# When and How to Autoconfigure

- Not all sites want any random node to be able to acquire an address and use it without authorisation
  - Auto-config does not enable that
    - it does make it easy
- DHCP used to assign addresses in IPv4, can implement policy
  - Auto-config has no policy
- Some users concerned about privacy
  - Every IPv6 address they use contains their MAC address
- Hence need alternative methods

# DHCPv6 & others

- DHCPv6 - very similar to DHCP
  - Managed or arbitrary address assignment
  - Can allocate addresses that contain EUI-64 or addresses 1 2 3 4 5 6 7 ...
- Manual Configuration
  - Always possible
  - 128 bits is a lot to configure correctly.
- RFC3041 Random Addresses
  - Avoids privacy concerns
  - Creates other problems

# Information for Nodes

- How does a node decide
  - what kind of address to use?
- From where does it discover
  - the network prefix (or prefixes)?
- If not using DHCP
  - how does it find a router to use?
- Router Advertisements
  - Router periodically sends
      - all this information
      - and more
    - to all nodes

# IPv6 Address Types

- Local Loopback
- Link Local Address
- Site Local Address
- Global Address
- IPv4 compatible Address
- Multicast Address

# Loopback Address

- ::1
  - :: (0::0) is the "unspecified address"
- IPv6 Address Notation
  - nnnn:nnnn::nnnn
    - The :: indicates as many 0's as are needed
    - Only one :: in any address
    - 16 bits in each other numeric block (between :'s)
- Loopback address means "this node"
  - 127.0.0.1 in IPv4

# Link Local Addresses

- Defined Prefix
  - FE80::/10
- Low 64 bits contain host identifier
  - fe80::210:a4ff:fe0f:41cf
  - fe80::1

# Link Local Addresses

- These addresses can be used to communicate with other nodes on the same link
  - Routers do not route packets containing link local addresses
  - Useful for all on-link communications
    - eg: router advertisement
  - Used where off link addressing is incorrect
    - eg: redirect
    - eg: Neighbour Discovery

# Site Local Addresses

- Defined Prefix
  - FEC0::/10
- Used for communications with a site
  - "site" can mean whatever is appropriate
  - Often a company/university/...
- Packets using these addresses are not forwarded beyond the boundaries of the site
  - FECr:rrrr:rrrr:SSSS:EUI-64
- Being deprecated

# Local Addresses

- Defined Prefix
  - FC00::/7
- Unroutable address
  - Not useful to reach random destination
- Possibly unique
  - some would say probably
  - some would hope certainly
  - nothing enforces uniqueness
    - no way to test either
- `FCaa:aaaa:aaaa:SSSS:EUI-64`
  - Assigned by number authority
- `FDrr:rrrr:rrrr:SSSS:EUI-64`
  - Generated by random number generator

# Global Addresses

- 48 bit site prefix
- 16 bit subnet number
- 64 bit EUI-64

| FP | Site Prefix | Subnet | Interface Identifier |
|---|---|---|---|

- FP: Format Prefix (3 bits: 001)
  - 001 010 011 100 101 110
    - All except 000 and 111
- Prefix: Site Identification (45 bits)
  - Internal aggregation boundaries exist
- Subnet: Network within site (16 bits)

# Old Global Addresses

| FP | TLA | NLA | SLA | Interface Identifier |
|---|---|---|---|---|

- TLA: Top Level Aggregator (13 bits)
  - 8192 globally visible prefixes
    - For each FP
      - For FP 001 anyway
- NLA: Next Level Aggregator (32 bits)
  - Expected to be further sub-divided
- SLA: Subnet (16 bits)
  - (Site Level Aggregator)
- Too inflexible
  - now no fixed boundaries

# IPv4 Compatible Addresses

- ::a.b.c.d
  - 96 bits of all 0
  - followed by an IPv4 address (32 bits)
- ::FFFF:a.b.c.d
  - 80 bits of all 0
  - followed by FFFF (16 bits)
  - then IPv4 address (32 bits)
- First form
  - used to number any node that has IPv6
  - and also has an IPv4 address
    - IPv4 compatibility mode
- Second form
  - used to number all IPv4 nodes that have no IPv6

# IPv4 Compatible Addresses

- ::a.b.c.d
- ::FFFF:a.b.c.d
- Packet translation is possible
  - These addresses generate the same
  - transport protocol checksums
  - as the IPv4 addresses they represent
- Used for API purposes only
  - Not routed on IPv6 network

# IPv4 Mapped Addresses

- 2002:a.b.c.d:SLA:EUI-64
  - Cannot be written this way
  - Must use
    - 2001:AABB:CCDD:SLA:EUI-64
  - a.b.c.d must be a global IPv4 address
- Any site with an IPv4 address
  - can use this as an IPv6 prefix
    - IPv4 internet is the IPv6 backbone

# Multicast Addresses

- FF00::/8
  - NB: this is not FF::/8
    - That would be 00FF::/8 or 00::/8

| FF | Flags | Scope | Group Identifier |
|---|---|---|---|
| 8 | 4 | 4 | 112 |

- Flags: 4 bits, only 1 assigned
  - T:   T==0 ==> Well Known Multicast Address
  -         T==1 ==> Transient Multicast Address
- Scope: 4 bit numeric field
  - 1   Node Local
  - 2   Link Local
  - 5   Site Local
  - 8   Organisation Local
  - E   Global

FF02::1- All Nod

Rest unassigned or rese