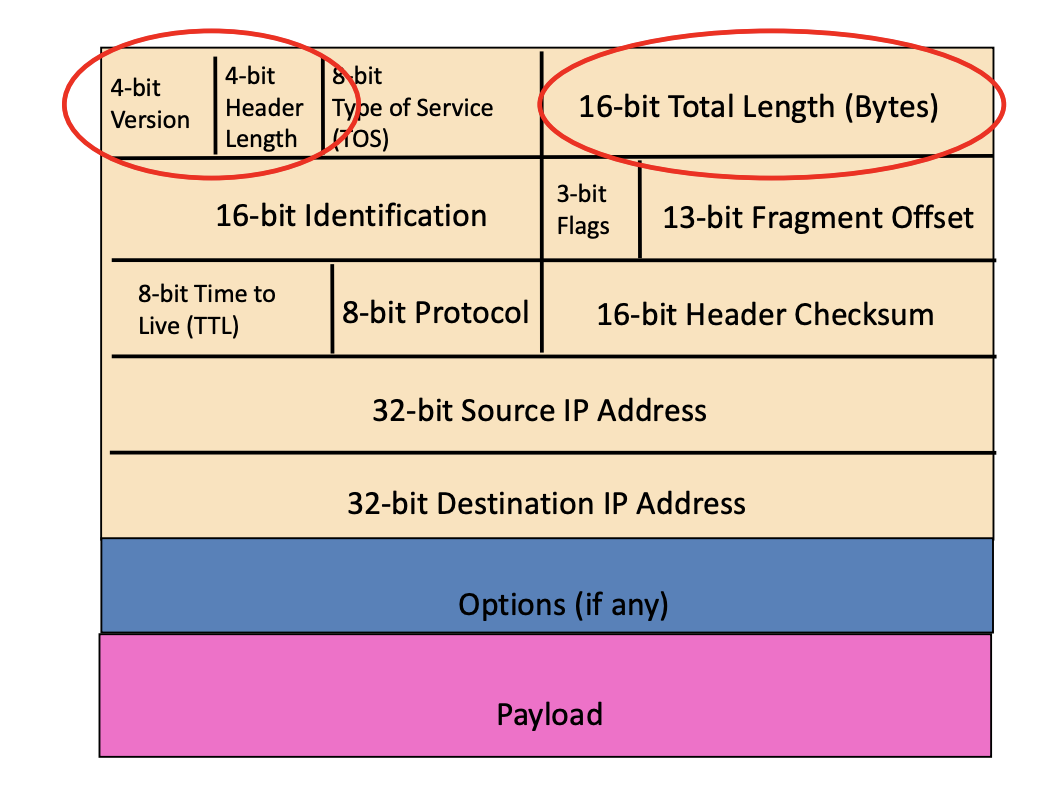
***The NETWORK LAYER***

**Overview**

* Routing Protocols
* IP protocol
* ICMP protocol

***Internet Protocol (IP)***

**IP Packet Structure**

****

* Version Number:
  + Indicates version of IP protocol
  + Typically, ‘4’ for IPv4
* Header Length:
  + Number of 32-bit words in header
  + Typically, ‘5’ for a 20 byte IPv4 header
  + Can be more when IP options used
* Total Length:
  + Total number of bytes in the packet
* 8-bit Protocol:
  + Identifies higher level Transport protocol (TCP/UDP)
  + Important for demultiplexing at receiving host
    - It’ll know whether to continue to send the packet on TCP or UDP
* Time to Live (TTL):
  + Actually indicates the max number of counts
  + If you set TTL to 4, the 5th router would actually drop the packet

**Preventing Router Loops using TTL**

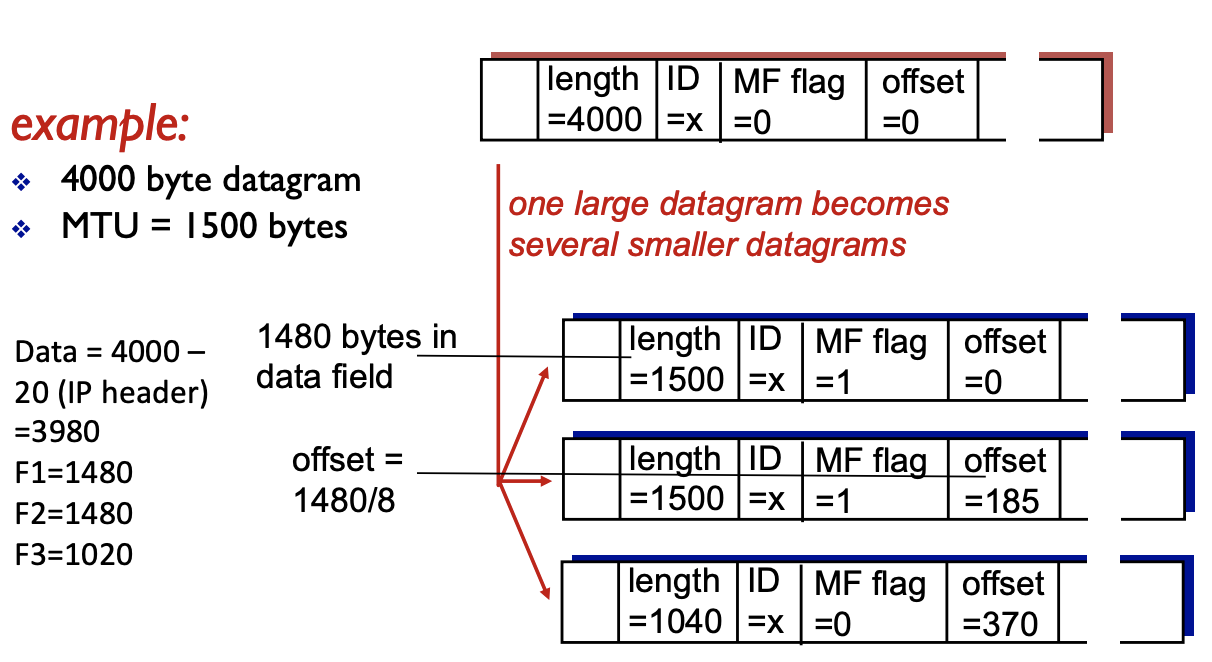
* Problem:
  + Forwarding loops cause packets to cycle and eventually consume all capacity
* TTL can be set so that once the TTL limit is reached, it’ll drop out and stop the loop

**Preventing Header Corruption using Checksum**

* Checksum only applies to the header for IP and NOT the payload
  + This is because TCP already covered the payload with its checksum, so IP doesn’t have to
* IP checksum recalculated at each router
  + This is because TTL changes between routers so the checksum needs to be changed too
  + If the checksum addition is wrong then the packet is discarded

***IP Fragmentation***

* Network links have a Maximum Transfer Size (MTU):
  + Maximum frame a link can carry
  + Different types of links have different MTUs
* A Large IP datagram has to be fragmented if its bigger than the MTU
  + Reassembled at final destination (NEVER AT THE ROUTER)
  + IP header bits are used to identify and order related fragments



* ID Flag:
  + ID of the original datagram
  + All subsequent fragments if the datagram would share the same ID
* MF Flag:
  + More Fragments flag
  + All fragments except the last will have 1
  + Last will be 0 because there are no more fragments
* Offset:
  + If offset is 0 and MF is 1, it means it is the FIRST fragment
  + Offset size is actual data (1480) / 8
* Length:
  + Data is really split into MTU – 20 (header size)
  + So 4000 = 3980 bytes
    - F1 = 1480
    - F2 = 1480
    - F3 = 1020

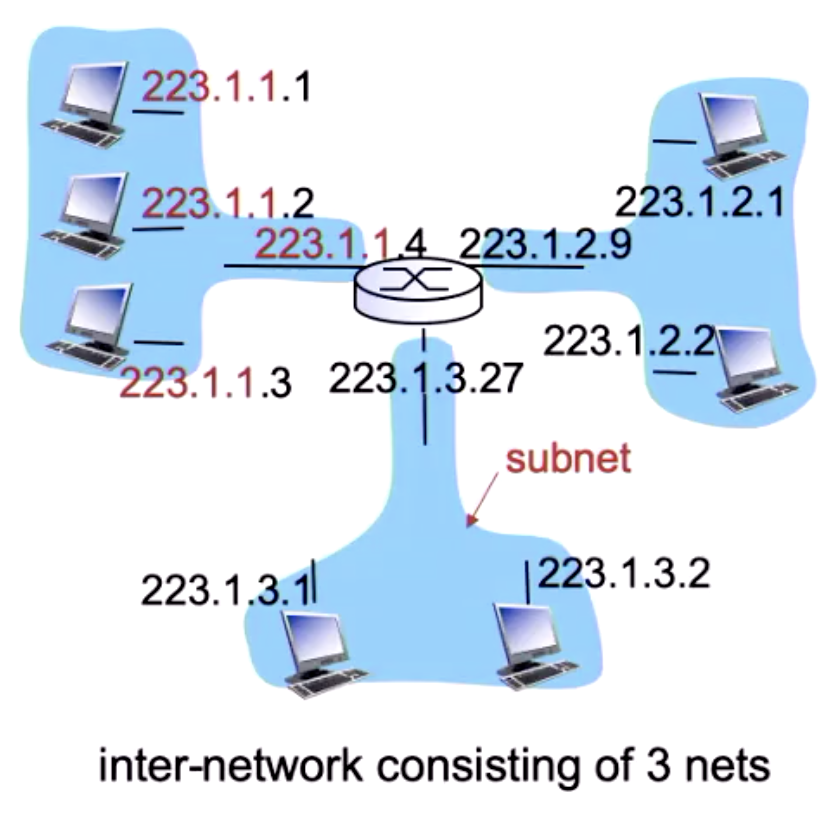
Receiving host uses MF, Offset, and ID to reassemble the datagram

**Malicious use of fragments**

* Sending two fragments with ‘overlapped offsets’ which some OS can’t handle and fucks the whole system lmao

**Overall Overhead:**

* 20 bytes TCP header
* 20 bytes IP header
* App layer overhead

***IPv4 Addressing***

IP Address:

* 32-bit identifier for host-router interface
* Every device MUST have an IP address
* For devices on the same network, only the last digit tends to be different

Interface:

* Connection between host/router and physical link
  + Routers typically have multiple interfaces
    - Example on right has 3
  + Hosts typically only have 1 or 2 interfaces

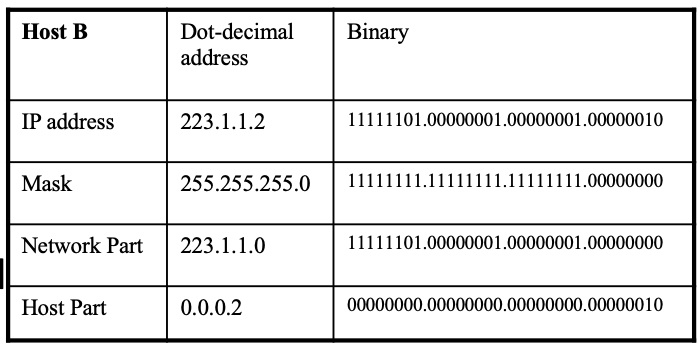
**Networks!**

What’s a network?

* Device interfaces with same network part of the IP address
* Devices in the same network can reach each other without going through the router

Masking

* Used in conjunction with network address to extract the Network part and host part of the address



* Bitwise AND operator used to produce network part and Host part

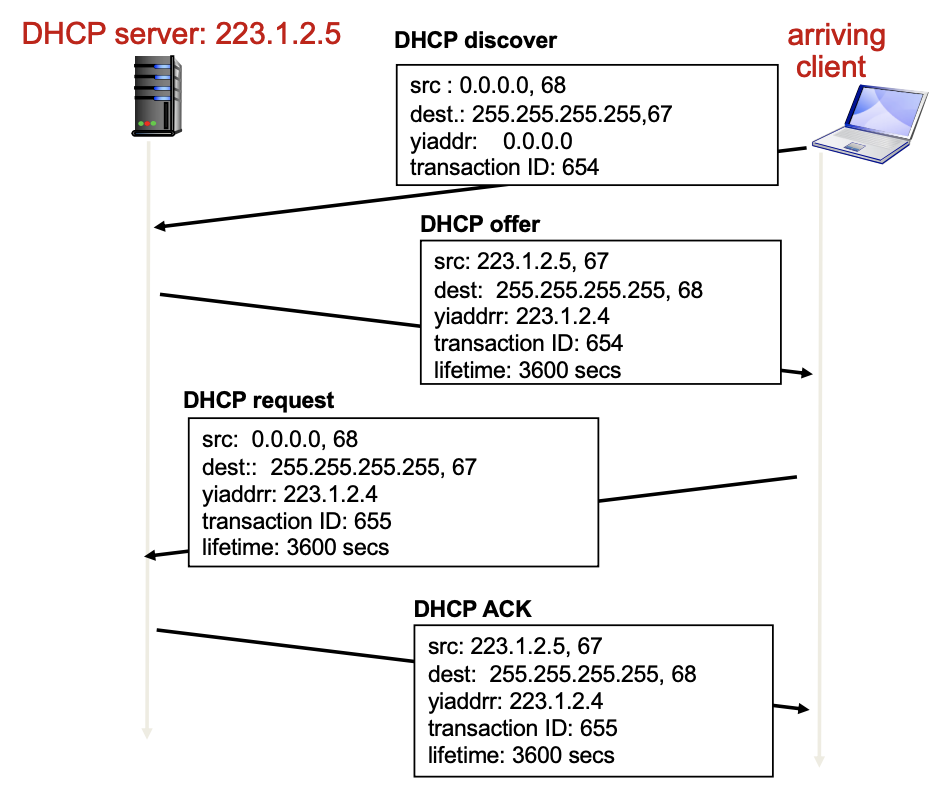
***Classless InterDomain Routing (CIDR)***

* Used in todays addressing
* Subnet addresses are of arbitrary length
* Address format: a.b.c.d/x
  + where x is # bits in subnet portion of addresses

***Receiving IP Addresses***

**Dynamic Host Configuration Protocol (DHCP)**

* Allows host to *dynamically* obtain its IP address from network server when it joins the network
* Allows reuse of addresses
  1. Only holds address while device is connected
* Overview of DHCP
  1. Host/device broadcasts ‘DHCP discover’ message
  2. DHCP server responds with ‘DHCP offer’ message
  3. Host requests IP with ‘DHCP request’ message
  4. DHCP server gives address: ‘DHCP Ack’ message



**Range of addresses DHCP servers can assign**

* If theres a subnet 129.94.100.0/25
  + We cannot use the network and broadcasting addresses:
    - 129.94.100.0 && 129.94.100.127
  + Therefore, DHCP can only use 129.94.100.1 🡪 129.94.100.126

***NETWORK ADDRESS TRANSLATION (NAT)***

***Private IP addresses***

* 10.0.0.0/8
* 172.16.0.0/12
* 192.168.0.0/16

Features of private IPs

* Private IP’s cannot be routed
* Typically used for NAT (Network Address Translation)

**Routing Protocols: Intra-Domain Routing (Link State / Distance Vector) and Inter-Domain Routing**

A routing protocol determines the end-to-end path of packets through the network.

The forwarding table determines the local forwarding at this router.

**Autonomous Systems (AS)** or **Domains** is a region of a network under a single admin authority. E.g. an ISP is an AS.

Internet routing works as two levels:

1. **Intra-Domain Routing Protocol:** AS Establishing routes within its own AS / domain.
   1. Single admin, so no policy decisions are needed. Performance > Policy.
   2. Examples of intra-domain routing:  
      Link State 🡪 **Open Shortest Path First (OSPF)**Distance Vector 🡪 **Routing Information Protocol (RIP)**
2. **Inter-Domain Routing Protocol**: AS Establishing routes between other AS / domains
   1. Admin wants control over routing in network + who routes through its network. Policy may > Performance.
   2. Examples of inter-domain routing:  
      Path Vector 🡪 **Border Gateway Protocol (BGP)**

**Link State Routing (Global)**: All routers have the complete topology and maintain / know the cost of each link in the network.

* How it works: (1) **Link State Advertisement (LSA) Flooding** (2) **Path calculation with Djikstra’s**
  + When receiving a new Link State msg, the router forwards it to all neighbours except one that sent the msg.
  + Routers keep a local copy so they don’t forward previously seen LSA’s.
  + Eventually, each node learns the entire network topology + can use Djikstra’s to compute shortest path.
* Eventually, each node learns entire network.
* Characteristics
  + Connectivity / cost changes are flooded to all routers in the network.
  + Converges quickly (less consistency, looping)
  + Limited network sizes, otherwise it will be too costly.
* Challenges:
  + Packet Loss / Out-of-order packets (solved with ACKs, Retransmissions, Seq Numbers, TTL for packets)
  + Scalability: # Messages to flood **O(N\*E)** where N = #nodes E = #edges | Djikstra’s **O(N2)**  
    # entries in topology database **O(E)** | # entries in forwarding table **O(N)**
  + Transient Disruptions / Infinite Loop problems: Inconsistent link-state database, as some routers know about failures before others. Shortest path is not always consistent, which can cause transient / infinite loops.
  + Oscillations: Costs can change around continuously. For given new costs 🡪 new route 🡪 new costs and so on.

**Distance Vector Routing (Decentralised)**: Routers only know its neighbours + link cost to neighbours.

* How it works:  
  (1) Each router initialises its DV table based on link costs to immediate neighbours + sends its DV to the neighbours.  
  (2) Neighbours process the DV and repeats STEP #1 until the iterative process converges to a set of shortest paths.  
  (3) Each node then waits for changes in their local link cost or msg from neighbours.  
  (4) If change occurs 🡪 recompute costs in DV and notify neighbours if anything changes.
* Initial state: best 1-hop paths | one simultaneous round = best 2-hop | k simultaneous rounds = best (K+1)-hop paths
* Characteristics:
  + Cost changes are iterative, exchanges info from neighbour to neighbour.
  + Requires multiple rounds to converge
  + Scales to large networks.
* **Counting to Infinity Problem (“bad news travels slowly”)**: Usually occurs when a node becomes broken.
  + Because of a broken link, nodes keep incorrectly updating their DV table and increasing cost for the broken link until the updates slowly propagates through the network and eventually reaches infinity.
* **Poisoned Reverse Rule** is a method to avoid the Count to Infinity Problem.
  + Routers actively advertise certain links as unreachable (cost=infinity). However, this will significantly increase the number of routing announcements made in the network.

Comparison of Link State vs. Distance Vector

|  |  |  |
| --- | --- | --- |
|  | **Link State** | **Distance Vector** |
| **Message Complexity** | N nodes / E edges = O(N\*E) messages sent | Exchange between neighbours only. |
| **Speed of Convergence** | O(N2) algorithm | relatively fast | Convergence time varies  Count to Infinity / Routing Loops may occur |
| **Robustness** | LS node can advertise incorrect LINK cost.  Each node computes only its own table. | DV node can advertise incorrect PATH cost.  Each node’s table is used by others, errors propagate through the network. |

**Problems with Distance Vector:**

* Most problems caused by slow convergence of routers or converging on wrong info
* Convergence:
  + The **time** in which all routers come to an agreement about the best paths in network
  + Reacts rapidly to good news (finding low-cost path) and slowly to bad news (cost increase)
    - Bad news would slowly iterate and can create the ‘count to infinity’ problem

***ICMP: Internet Control Message Protocol***