15-441/641: Computer Networks
The Internet Protocol

15-441 Spring 2019
Profs **Peter Steenkiste** & Justine Sherry

Fall 2019
https://computer-networks.github.io/sp19/

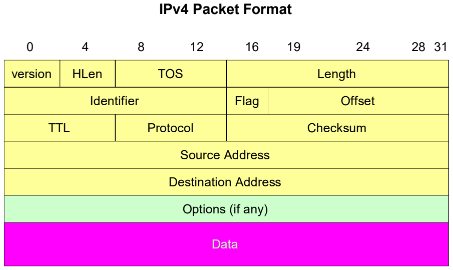
Carnegie Mellon University

Outline

- The IP protocol
  - IPv4
  - IPv6
- Tunnels

IP Service Model

- Low-level communication model provided by Internet
- Datagram: each packet is self-contained
  - All information needed to get to destination
  - No advance setup or connection maintenance
  - Analogous to letter or telegram

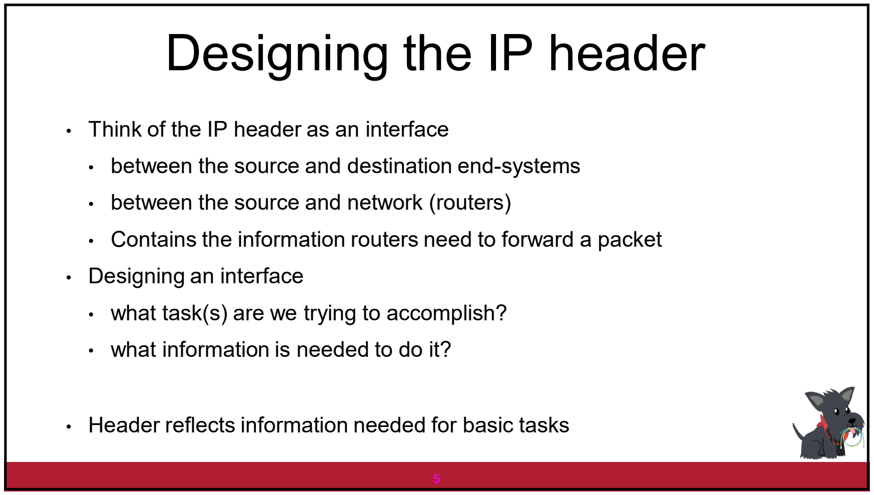


IPv4 Packet Format

version	HLen	TOS	Length
Identifier		Flag	Offset
TTL	Protocol	Checksum	
Source Address			
Destination Address			
Options (if any)			
Data			

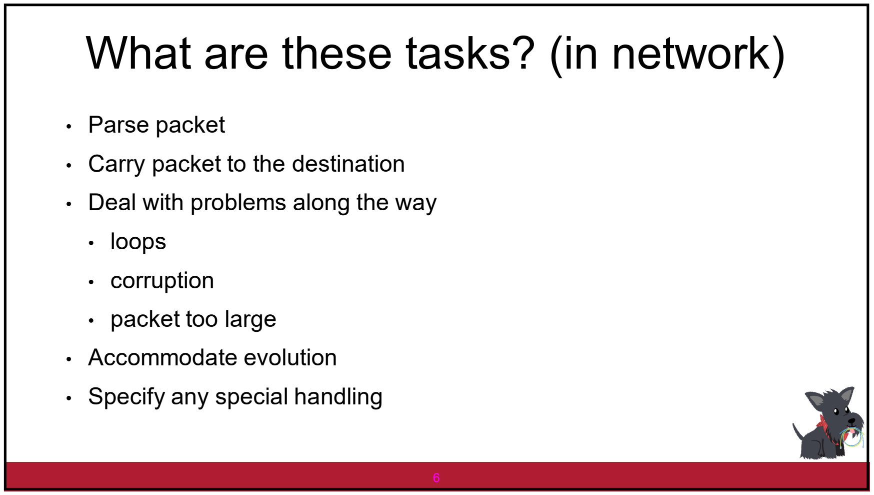
IP Delivery Model

- *Best effort service*
  - Network will do its best to get packet to destination
- Does NOT guarantee:
  - Any maximum latency or even ultimate success
  - Informing the sender if packet does not make it
  - Delivery of packets in same order as they were sent
  - Just one copy of packet will arrive
- Implications
  - Scales very well (really, it does)
  - Higher level protocols must make up for shortcomings
    - Reliably delivering ordered sequence of bytes → TCP
  - Some services not feasible (or hard)
    - Latency or bandwidth guarantees



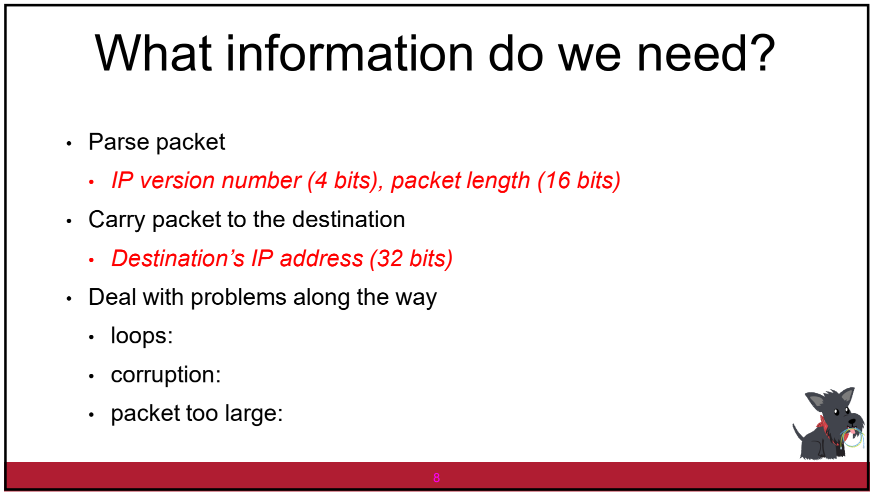
Designing the IP header

- Think of the IP header as an interface
  - between the source and destination end-systems
  - between the source and network (routers)
  - Contains the information routers need to forward a packet
- Designing an interface
  - what task(s) are we trying to accomplish?
  - what information is needed to do it?
- Header reflects information needed for basic tasks



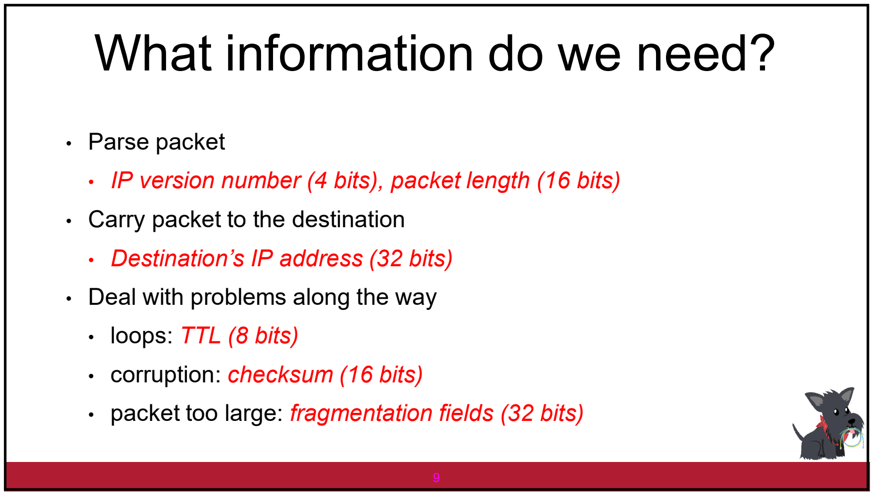
What are these tasks? (in network)

- Parse packet
- Carry packet to the destination
- Deal with problems along the way
  - loops
  - corruption
  - packet too large
- Accommodate evolution
- Specify any special handling



What information do we need?

- Parse packet
  - *IP version number (4 bits), packet length (16 bits)*
- Carry packet to the destination
  - *Destination’s IP address (32 bits)*
- Deal with problems along the way
  - loops:
  - corruption:
  - packet too large:

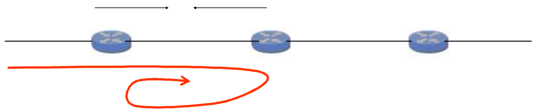


What information do we need?

- Parse packet
  - *IP version number (4 bits), packet length (16 bits)*
- Carry packet to the destination
  - *Destination’s IP address (32 bits)*
- Deal with problems along the way
  - loops: *TTL (8 bits)*
  - corruption: *checksum (16 bits)*
  - packet too large: *fragmentation fields (32 bits)*

Preventing Loops (TTL)

- Forwarding loops cause packets to cycle for a very looong time
  - left unchecked would accumulate to consume all capacity



- Time-to-Live (TTL) Field (8 bits)
  - decremented at each hop, packet discarded if reaches 0
  - …and “time exceeded” message is sent to the source

Header Corruption (Checksum)

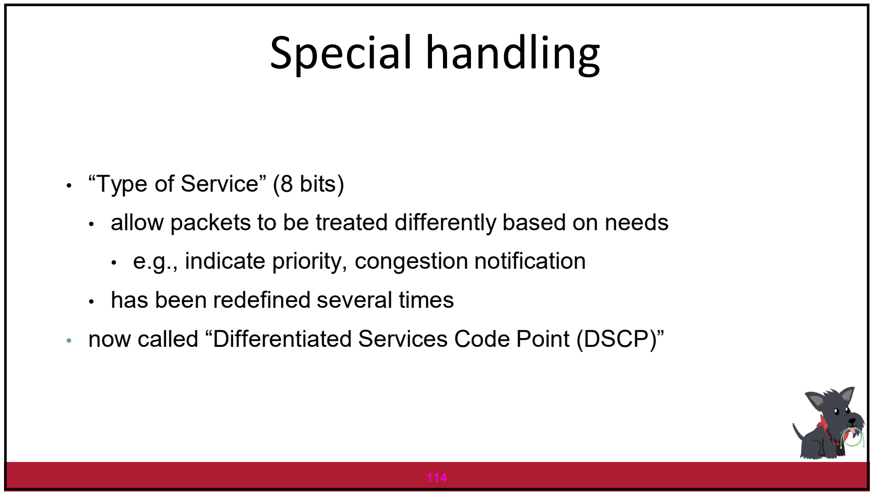
- Checksum (16 bits)
  - Particular form of checksum over packet header
- If not correct, router discards packets
  - So it doesn’t act on bogus information
- Checksum recalculated at every router
  - Why?

Fragmentation

- Every link has a “Maximum Transmission Unit” (MTU)
  - largest number of bits it can carry as one unit
- A router can split a packet into multiple “fragments” if the packet size exceeds the link’s MTU
- Must reassemble to recover original packet
- Will return to fragmentation shortly…

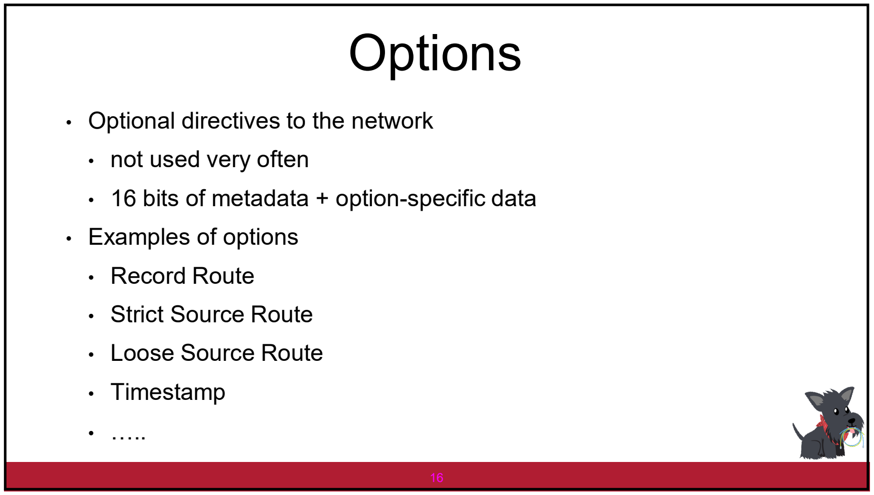
What information do we need?

- Parse packet
  - *IP version number (4 bits), packet length (16 bits)*
- Carry packet to the destination
  - *Destination’s IP address (32 bits)*
- Deal with problems along the way
  - *TTL (8 bits), checksum (16 bits), fragmentation (32 bits)*
- Accommodate evolution
  - *version number (4 bits) (+ fields for special handling)*
- Specify any special handling



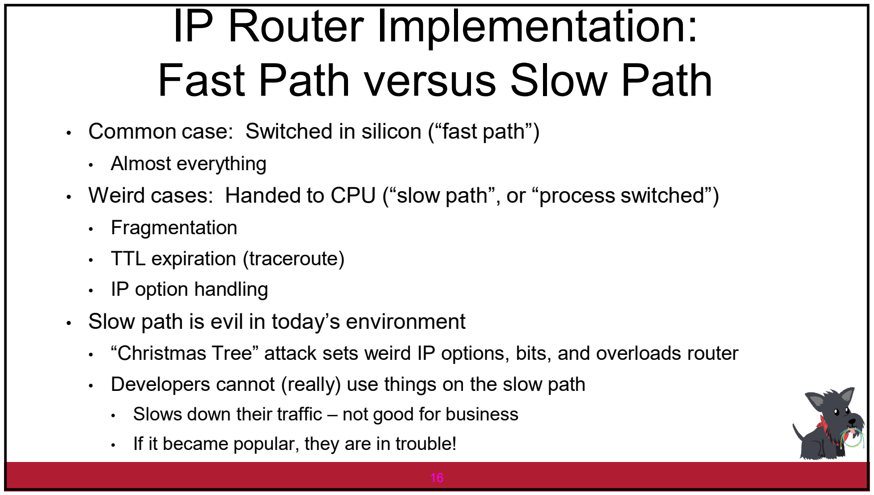
Special handling

- “Type of Service” (8 bits)
  - allow packets to be treated differently based on needs
    - e.g., indicate priority, congestion notification
  - has been redefined several times
- now called “Differentiated Services Code Point (DSCP)”



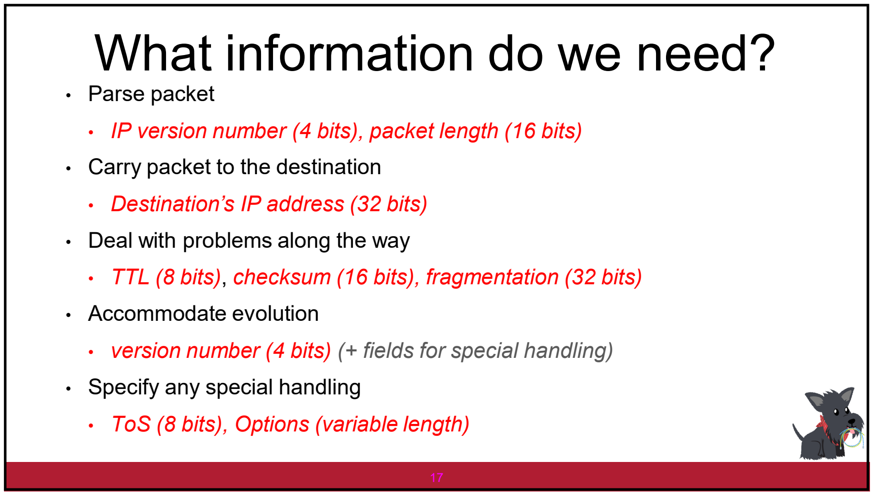
Options

- Optional directives to the network
  - not used very often
  - 16 bits of metadata + option-specific data
- Examples of options
  - Record Route
  - Strict Source Route
  - Loose Source Route
  - Timestamp
  - …..



IP Router Implementation: Fast Path versus Slow Path

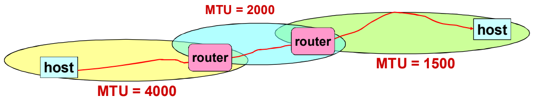
- Common case: Switched in silicon (“fast path”)
  - Almost everything
- Weird cases: Handed to CPU (“slow path”, or “process switched”)
  - Fragmentation
  - TTL expiration (traceroute)
  - IP option handling
- Slow path is evil in today’s environment
  - “Christmas Tree” attack sets weird IP options, bits, and overloads router
  - Developers cannot (really) use things on the slow path
    - Slows down their traffic – not good for business
    - If it became popular, they are in trouble!



What information do we need?

- Parse packet
  - *IP version number (4 bits), packet length (16 bits)*
- Carry packet to the destination
  - *Destination’s IP address (32 bits)*
- Deal with problems along the way
  - *TTL (8 bits), checksum (16 bits), fragmentation (32 bits)*
- Accommodate evolution
  - *version number (4 bits) (+ fields for special handling)*
- Specify any special handling
  - *ToS (8 bits), Options (variable length)*

IP Fragmentation

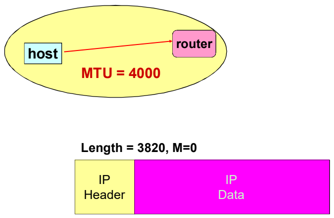


- Every network has own Maximum Transmission Unit (MTU)
  - Largest IP datagram it can carry within its own packet frame
    - E.g., Ethernet is 1500 bytes
  - Don't know MTUs of all intermediate networks in advance
- IP Solution
  - When hit network with small MTU, router fragments packet
  - Destination host reassembles the paper – why?

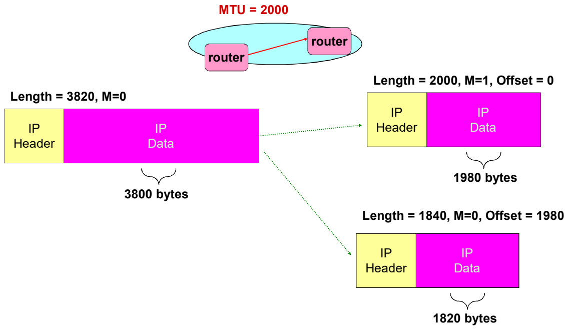
Fragmentation Related Fields

- Length
  - Length of IP fragment
- Identification
  - To match up with other fragments
- Flags
  - Don't fragment flag
  - More fragments flag
- Fragment offset
  - Where this fragment lies in entire IP datagram
  - Measured in 8 octet units (13 bit field)

IP Fragmentation Example #1



IP Fragmentation Example #2



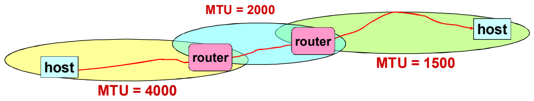
Fragmentation is Harmful

- Uses resources poorly
  - Forwarding costs per packet
  - Best if we can send large chunks of data
  - Worst case: packet just bigger than MTU
- Poor end-to-end performance
  - Loss of a fragment
- Path MTU discovery protocol → determines minimum MTU along route
  - Uses ICMP error messages
- Common theme in system design
  - Assure correctness by implementing complete protocol
  - Optimize common cases to avoid full complexity

Internet Control Message Protocol (ICMP)

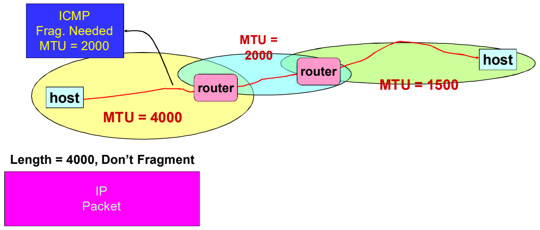
- Short messages used to send error & other control information
- Some functions supported by ICMP:
  - Ping request /response: check whether remote host reachable
  - Destination unreachable: Indicates how packet got & why couldn't go further
  - Flow control: Slow down packet transmit rate
  - Redirect: Suggest alternate routing path for future messages
  - Router solicitation / advertisement: Helps newly connected host discover local router
  - Timeout: Packet exceeded maximum hop limit
- How useful are they functions today?

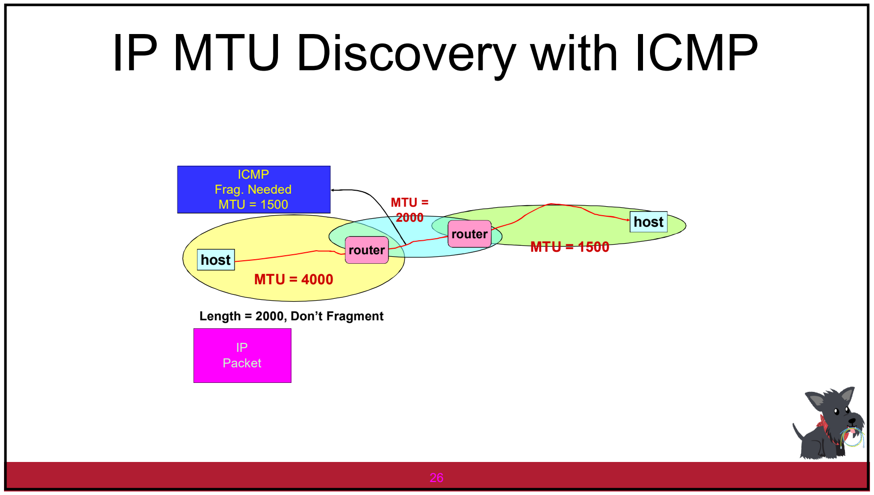
IP MTU Discovery with ICMP



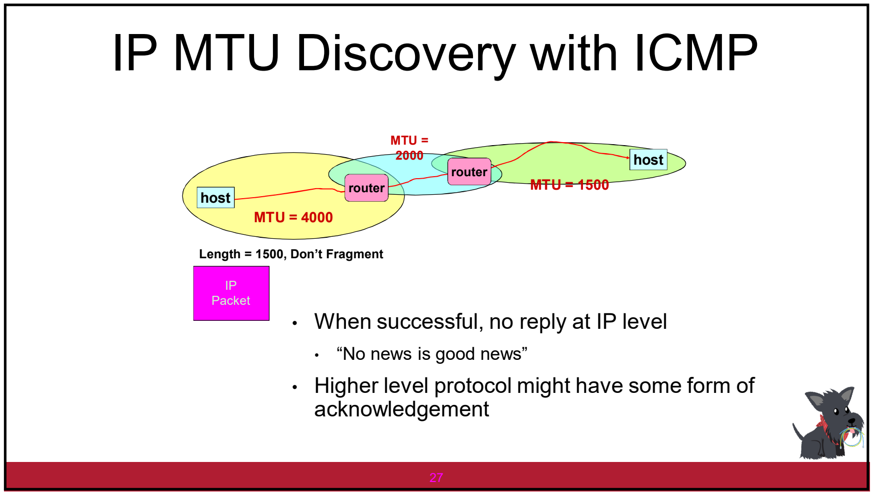
- Typically send series of packets from one host to another
- Typically, all will follow same route – routes are stable for minutes at a time
- Makes sense to determine path MTU before sending real packets
- Operation: Send max-sized packet with "do not fragment" flag set
  - If a router encounters a problem, it will return ICMP message to the sender
    - "Destination unreachable: Fragmentation needed"
    - Usually indicates MTU problem encountered
- ICMP abuse? Other solutions?

IP MTU Discovery with ICMP



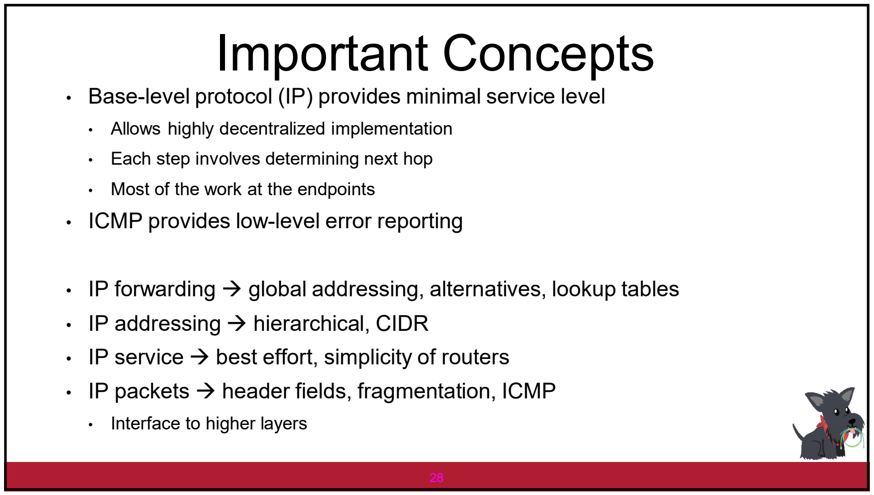


IP MTU Discovery with ICMP



IP MTU Discovery with ICMP

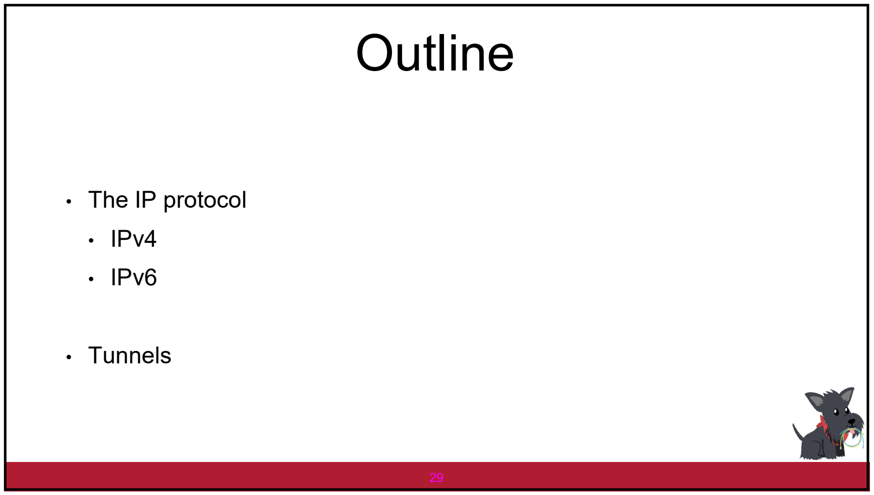
- When successful, no reply at IP level
  - "No news is good news"
- Higher level protocol might have some form of acknowledgement



Important Concepts

- Base-level protocol (IP) provides minimal service level
  - Allows highly decentralized implementation
  - Each step involves determining next hop
  - Most of the work at the endpoints
- ICMP provides low-level error reporting

- IP forwarding → global addressing, alternatives, lookup tables
- IP addressing → hierarchical, CIDR
- IP service → best effort, simplicity of routers
- IP packets → header fields, fragmentation, ICMP
  - Interface to higher layers



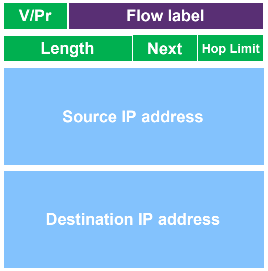
Outline

- The IP protocol
  - IPv4
  - IPv6

- Tunnels

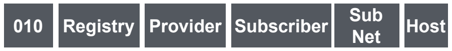
IPv6

- “Next generation” IP
- Most urgent issue: increasing address space.
  - 128 bit addresses
- Simplified header for faster processing:
  - No checksum (why not?)
  - No fragmentation (really?)
- Support for guaranteed services:
  - Priority and flow identifier
- Options handled as “next header”
  - reduces overhead of handling options



IPv6 Address Size Discussion

- Do we need more addresses? Probably, long term
  - Big panic in 90s: “We’re running out of addresses!”
  - Big worry: Devices. Small devices. Cell phones, toasters, everything.
- 128 bit addresses provide space for structure (good!)
  - Hierarchical addressing is much easier
  - Assign an entire 48-bit sized chunk per LAN – use Ethernet addresses
  - Different chunks for geographical addressing, the IPv4 address space,
  - Perhaps help clean up the routing tables - just use one huge chunk per ISP and one huge chunk per customer.

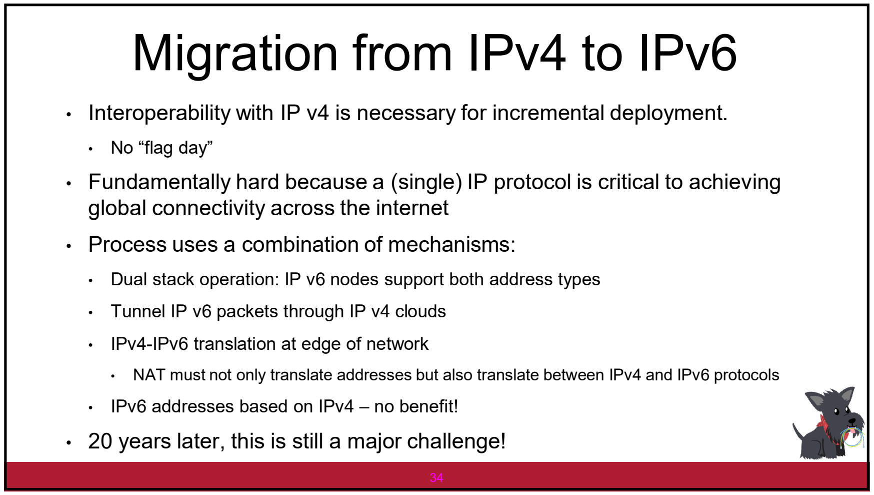


IPv6 Header Cleanup: Options

- 32 IPv4 options → variable length header
  - Rarely used
  - No development / many hosts/routers do not support
    - Worse than useless: Packets w/options often even get dropped!
  - Processed in “slow path”.
- IPv6 options: “Next header” pointer
  - Combines “protocol” and “options” handling
    - Next header: “TCP”, “UDP”, etc.
  - Extensions header: Chained together
  - Makes it easy to implement host-based options
  - One value “hop-by-hop” examined by intermediate routers
    - E.g., “source route” implemented only at intermediate hops

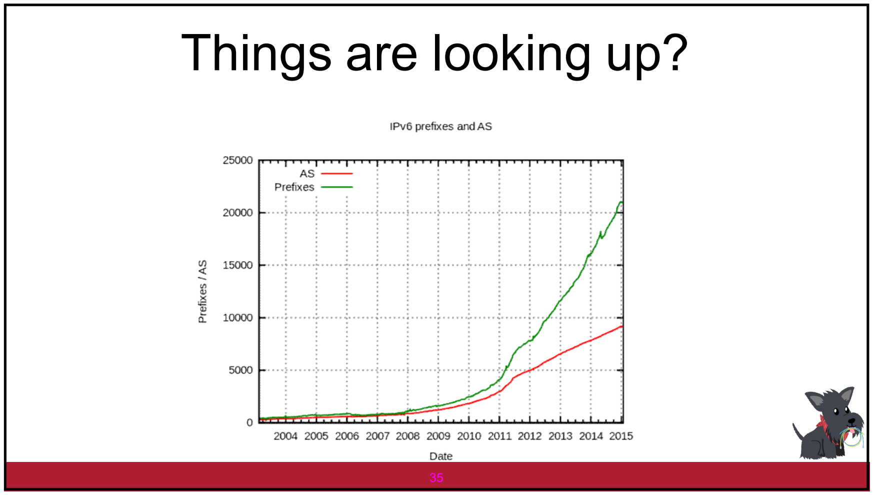
IPv6 Header Cleanup: “no”

- No checksum
  - Motivation was efficiency: If packet corrupted at hop 1, don’t waste b/w transmitting on hops 2..N.
  - Useful when corruption frequent, bandwidth expensive
  - Today: corruption is rare, bandwidth is cheap
- No fragmentation
  - Router discard packets, send ICMP “Packet Too Big” → host does MTU discovery and fragments
  - Reduced packet processing and network complexity.
  - Increased MTU a boon to application writers
  - Hosts can still fragment - using fragmentation header. Routers don’t deal with it any more.

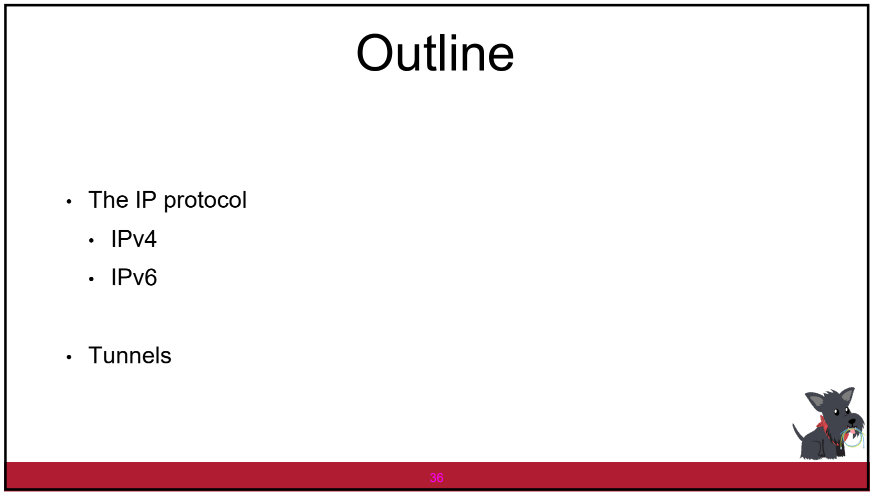


Migration from IPv4 to IPv6

- Interoperability with IP v4 is necessary for incremental deployment.
  - No “flag day”
- Fundamentally hard because a (single) IP protocol is critical to achieving global connectivity across the internet
- Process uses a combination of mechanisms:
  - Dual stack operation: IP v6 nodes support both address types
  - Tunnel IP v6 packets through IP v4 clouds
  - IPv4-IPv6 translation at edge of network
    - NAT must not only translate addresses but also translate between IPv4 and IPv6 protocols
  - IPv6 addresses based on IPv4 – no benefit!
- 20 years later, this is still a major challenge!

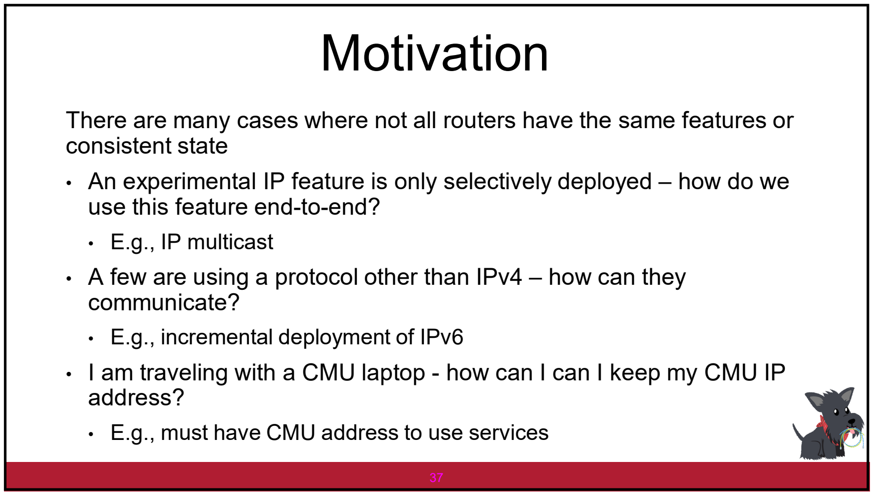


Things are looking up?



Outline

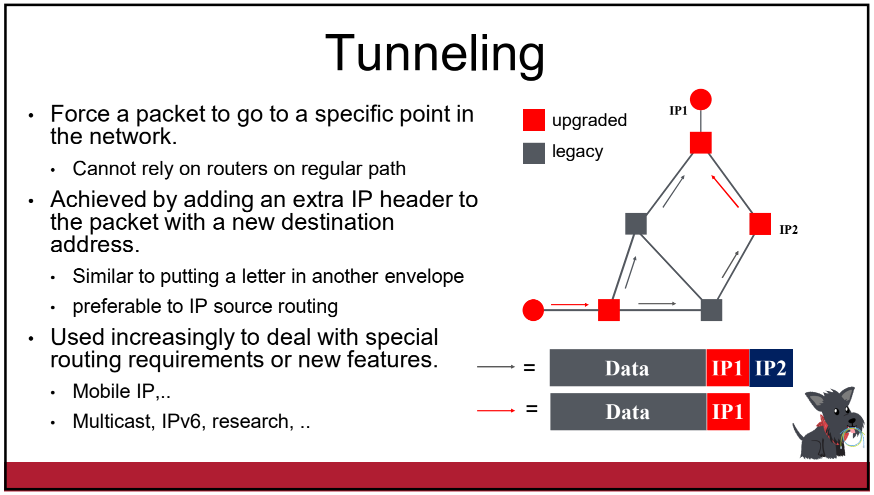
- The IP protocol
  - IPv4
  - IPv6
- Tunnels



Motivation

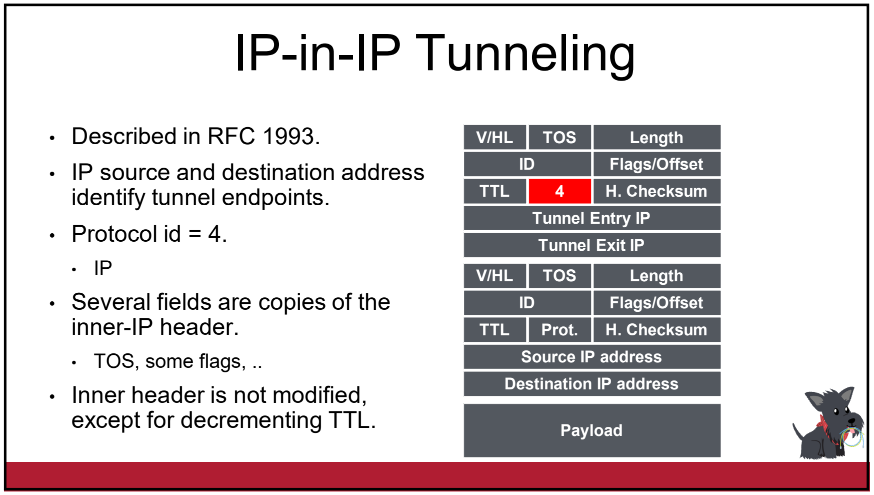
There are many cases where not all routers have the same features or consistent state

- An experimental IP feature is only selectively deployed – how do we use this feature end-to-end?
  - E.g., IP multicast
- A few are using a protocol other than IPv4 – how can they communicate?
  - E.g., incremental deployment of IPv6
- I am traveling with a CMU laptop - how can I can I keep my CMU IP address?
  - E.g., must have CMU address to use services



Tunneling

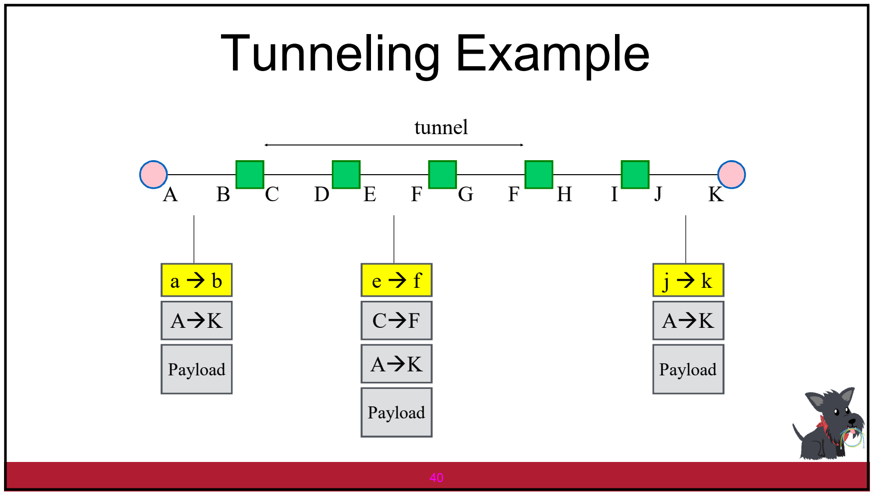
- Force a packet to go to a specific point in the network.
  - Cannot rely on routers on regular path
- Achieved by adding an extra IP header to the packet with a new destination address.
  - Similar to putting a letter in another envelope
  - preferable to IP source routing
- Used increasingly to deal with special routing requirements or new features.
  - Mobile IP,..
  - Multicast, IPv6, research, ..



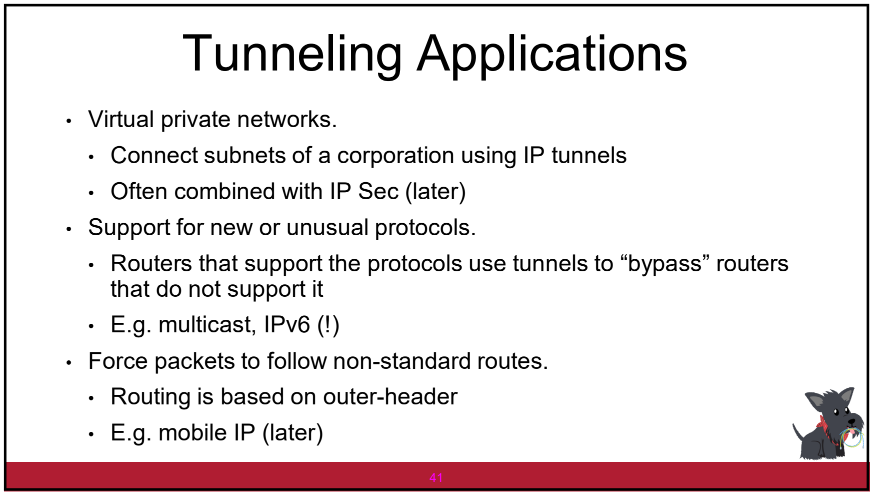
IP-in-IP Tunneling

- Described in RFC 1993.
- IP source and destination address identify tunnel endpoints.
- Protocol id = 4.
  - IP
- Several fields are copies of the inner-IP header.
  - TOS, some flags, ..
- Inner header is not modified, except for decrementing TTL.

V/HL	TOS	Length
ID		Flags/Offset
TTL	4	H. Checksum
Tunnel Entry IP		
Tunnel Exit IP		
V/HL	TOS	Length
ID		Flags/Offset
TTL	Prot.	H. Checksum
Source IP address		
Destination IP address		
Payload		



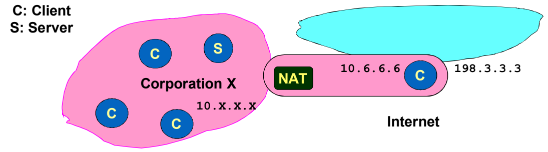
Tunneling Example



Tunneling Applications

- Virtual private networks.
  - Connect subnets of a corporation using IP tunnels
  - Often combined with IP Sec (later)
- Support for new or unusual protocols.
  - Routers that support the protocols use tunnels to “bypass” routers that do not support it
  - E.g. multicast, IPv6 (!)
- Force packets to follow non-standard routes.
  - Routing is based on outer-header
  - E.g. mobile IP (later)

Extending Private Network



- Supporting Road Warrior
  - Employee working remotely with assigned IP address 198.3.3.3
  - Wants to appear to rest of corporation as if working internally
    - From address 10.6.6.6
    - Gives access to internal services (e.g., ability to send mail)
- Virtual Private Network (VPN)
  - Overlays private network on top of regular Internet