49202 Communication Protocols

The Network Layer - Part 2

Daniel R. Franklin

Faculty of Engineering & IT

April 25, 2023

Network layer - continued

- Today we will extend our discussion of the network layer:
  - IPv4 fragmentation and reassembly
  - Private IP addresses
  - Path MTU discovery
  - ICMP
  - A critical IPv4 helper protocol - DHCP
  - IPv6

Fragmentation & reassembly

- Different networks may have different maximum transmission unit (MTU) sizes. This is the maximum payload size at Layer 2.
- If the MTU of network N is greater than that of network $N + 1$ in the end-to-end path, the router connecting them together will need to fragment packets which are too large to forward on network $N + 1$
- Fragments are re-assembled at the destination. Why?
  - Different fragments may (rarely) take different paths through the network - reassembly may only be possible at the destination
  - Routers would have more work to do, and would need to buffer until all fragments are reassembled - but the packet may then need to be fragmented in a later network!
- Consequence: IPv4 packets can only get smaller - possibly leading to inefficient operation of later networks
- Fragmentation is bad - better to try to avoid it in the first place.

Fragmentation & reassembly

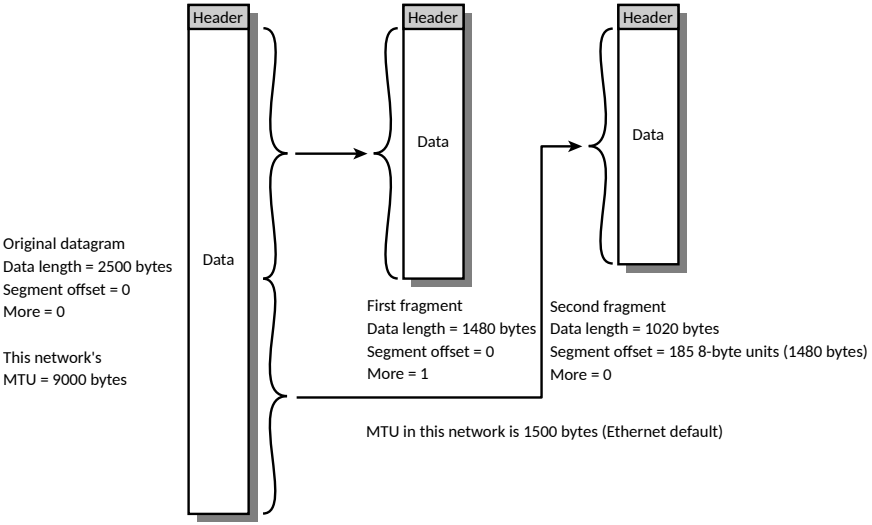
- In IPv4, fragmentation and reassembly make use of the following fields in the IP header:
  - Data Unit Identifier - meta-header comprising source and destination addresses + other protocol information
  - Data length - length of data field in bytes (sometimes called *octets*)
  - `offset` - position of start of this fragment
  - `more` flag - 1 = more segments to follow; 0 = this is the last, no segments to follow

Fragmentation & reassembly

So, if fragmentation is required:

- Make two or more *new* datagrams with (mostly) the same header
- Divide the payload into portions which are multiples of 8 bytes and, with the header, will fit within the MTU
- Set the data length to the length of new data field, set the `more` flag to = 1, `offset` field to length of preceding data
- Last packet has `more` = 0

IP datagram fragmentation



Why not avoid fragmentation in the first place?

- Fragmentation is **always** bad - it can lead to out-of-order arrivals and can only reduce the efficiency of the network
- In the majority of cases, many more than one IP datagram will flow between a source and destination. Fragmenting and reassembling all of these is a lot of work!
- An alternative approach is to prevent fragmentation in the first place by setting the **DF** flag in the IP header (or by using IPv6, which does not support fragmentation)
- What happens in this case?

Path MTU discovery

- When a router cannot forward a datagram because it is too large for the MTU of the next network and **DF** is set (or it is IPv6), it will drop the packet
- BUT it does not do so silently - instead, it sends an ICMP **Fragmentation Needed** error message back to the sender (or, for IPv6, ICMPv6 **Packet Too Big**)
- The sender then reduces the datagram size to fit within the MTU of the problematic network
- The process repeats if needed until the datagram can reach its destination
- Typically this will only require a few retransmission; in a many-datagram sequence this will be a win
- This is known as `Path MTU discovery` and is standard default practice on modern operating systems.

Internet Control Message Protocol

- We have talked about ICMP quite a lot at this point (and you've seen it in the lab). But what is it?
- ICMP is an essential helper protocol for debugging networks and reporting errors
- User-visible tools which utilise ICMP include `ping` and `traceroute` which are used to test reachability and explore the path taken by a packet through a network.
- It uses no transport layer protocol, sitting directly on top of IP.
- Versions exist for IPv4 and IPv6 (ICMPv6)

ICMP packet structure

- ICMP packets have an 8-byte header:
  - 1-byte ICMP packet type
  - 1-byte ICMP code (a subtype)
  - 2-byte checksum
  - 4-byte rest-of-header field (interpretation depends on the type/subtype)
- This is then followed by the data field; this includes the entire IP header of the packet which caused this ICMP message to be sent, plus at least the first 8 bytes of the payload.

ICMP packet types

- Many different ICMP (IPv4) packet types are defined. Examples include
  - Type 0 - echo (ping) reply. Code = 0.
  - Type 3 - destination unreachable. Code = 0-15, indicating the reason (e.g. 0 = network unreachable, 1 = host unreachable, 2 = protocol unreachable, 3 = port unreachable, 4 = fragmentation required etc.)
  - Type 8 - echo (ping) request. Code = 0.
  - Type 11 - time exceeded. Code 0 = TTL expired, 1 = fragment reassembly time exceeded
- Many ICMP packet types are now obsolete and are no longer used.
- ICMPv6 is very similar - obsolete types/codes are removed and additional entries have been added, but the structure is much the same.

Traceroute

- Traceroute is an interesting hybrid protocol used to identify the routers between a packet source and destination
- A UDP datagram or TCP segment is sent to a given destination with the IP TTL field set to 1
- The first router which receives this datagram decrements the TTL and then drops the packet
- When it does this, it sends back an ICMP **TTL expired** message tot he sender
- The sender then increases the TTL and sends the packet again
- This time, it is forwarded by the first router (setting TTL = 1), then it arrives at the final router (TTL = 0) where the packet is discarded and another ICMP **TTL expired** message is returned to the sender.
- The process repeats, each time with a higher initial TTL, until the packet arrives at its destination (or the destination is unreachable). Each iteration provides information about one more hop along the path.

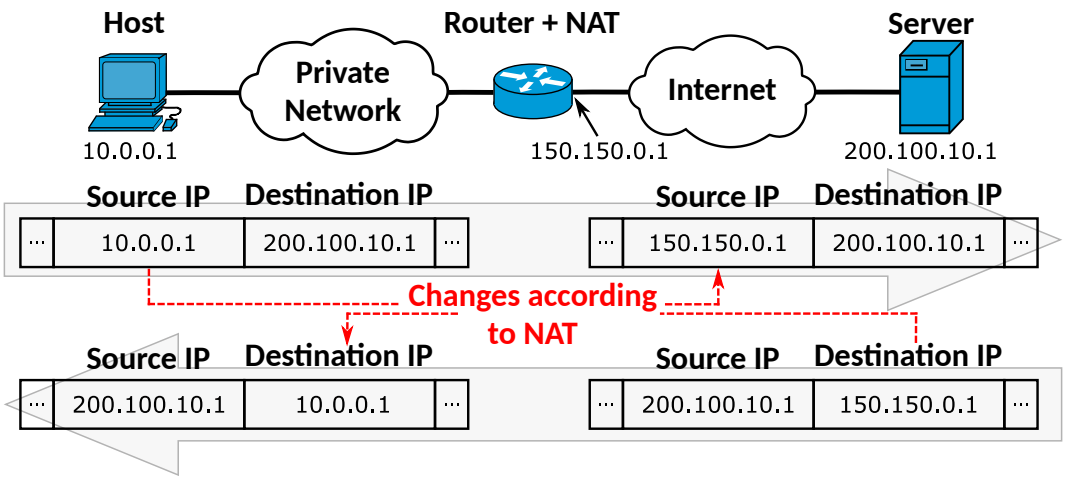
Private IP addresses and Network Address Translation (NAT)

- This is a relatively niche topic, so we will cover it briefly here before moving on to routing protocols
- Certain address ranges are reserved for *private use*:
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16
- These are not assigned to any one user, rather they can be used by anyone in their own network
- Typically you'll get an address in one of these ranges (or a subnet of these) from your home router - they are also commonly seen in enterprise wireless networks (e.g. at UTS).

Private IP addresses and NAT

- You *cannot* advertise a route to these destinations - so how are they useful?
- Your router can perform **network address translation** when forwarding a packet from the private network to the public Internet:
  - The **source IP address** is **changed** to the external (real) IP address of the router
  - The **source port** is mapped to a free port on the router's external interface
  - The router remembers the association using the 16-bit **Identification** field of the IPv4 header
- If an IP datagram comes back in reply, the router recognises the Identification field:
  - The **destination IP address** is translated back to the original private IP address (from the outgoing packet)
  - The **destination port** is mapped back to the original source port
- The process is largely transparent

Private IP addresses and NAT



Private IP addresses and NAT

- NAT allows a single real IPv4 address to be shared between a large number of private hosts
- It has delayed the need for IPv6 for decades - a big win for the Internet
- Now, some ISPs are even deploying it at the carrier level - carrier grade NAT
- NAT makes it difficult to host your own services at home (e.g. a public low-traffic web server, private VPN server etc.). CG-NAT makes it impossible.
- You *can* selectively externally-facing ports on your router to internal hosts (again with help from NAT).
- This is not possible on CG-NAT networks - but most customers don't care

Dynamic Host Configuration Protocol (DHCP)

- UDP-based protocols which is used for client network autoconfiguration. Clients can automatically obtain:
  - IP address
  - Netmask
  - Gateway
  - DNS server, search path and suffix
- DHCP also can distribute a variety of arbitrary binary data, such as a static routing table. Not all clients support these optional features.

Dynamic host configuration protocol (DHCP)

- The DHCP server listens on UDP port 67
- A client wishing to join a network DHCP starts by *broadcasting* a **DHCPDISCOVER** UDP/IP datagram in a broadcast frame:
  - Ethernet: source=sender's MAC; destination=FF:FF:FF:FF:FF:FF (broadcast destination)
  - IP: source=0.0.0.0; destination=255.255.255.255 (limited broadcast destination)
  - UDP: source port=68; destination port=67
  - Payload: many fields (see next slide)

Dynamic host configuration protocol (DHCP)

- Payload:
  - 4 8-bit bytes: OP (1 = request, 2 = reply), HTYPE (6 for Ethernet/WiFi), HLEN (6 for Ethernet/WiFi), HOPS (always zero unless being relayed via another network)
  - 32-bit XID (transaction identifier - so the client can match the response to its request)
  - 2-byte 'time in seconds' since the first request of this transaction
  - 2-byte flags field: only bit 1 is used (set to 1 if the client does not yet know its own IP address)
  - 32-bit CIAddr, YIAddr, SIAddr, GIAddr: client address if we already have one (otherwise 0), address being assigned to the client by the server (0 if not yet specified), address of the DHCP server (if known, otherwise 0) and address of the relay (gateway - note, NOT for routing, only for DHCP).
  - 16-byte CHAddr: client hardware address - layer 2 address of the client, needed in case the request/response is being relayed
  - 64-byte SName: optional - may be set by the server to some arbitrary text OR can be used to carry options(!)

Dynamic host configuration protocol (DHCP)

- If a DHCP server sees a **DHCPDISCOVER** message, the server reserves an address from its address pool (OR it may be statically associated with the Layer 2 address of the client)
- It will send back a **unicast DHCPOFFER** response, which includes the server IP address (SIAddr), IP address (YIAddr), subnet mask (in an option), default router (i.e. gateway, also in an option), duration of the lease (in seconds, another option) and list of DNS servers (option). The XID field will match the one set by the client.
- If the client accepts the offer, broadcast a **DHCPREQUEST** reply with the selected server's ID in the SIAddr field. This is because there may be more than one offer; the client therefore accepts only one, and other servers will withdraw their offers.
- The server sends back a unicast **DHCPACK** acknowledging the request; CIAddr is the assigned address, SIAddr is the server's address.
- Before it starts to use the allocated address, the client sends an ARP request

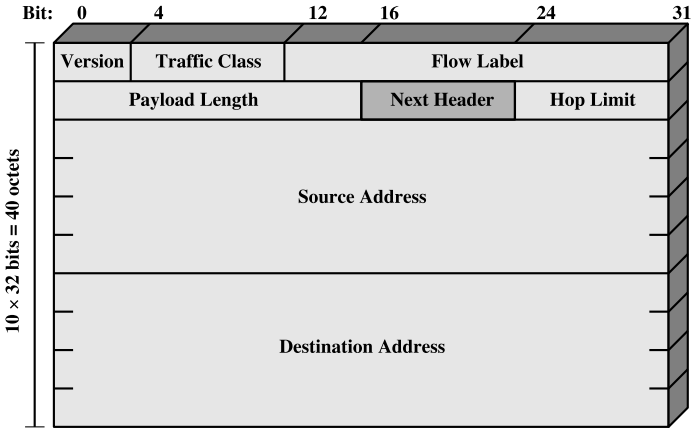
IPv6

- One of the problems with IPv4 is the size of its address space: 32 bits = 4294967295 addresses (minus the invalid ones...)
- While NAT does help with this problem, it can create others
  - It is non-trivial to communicate bidirectionally over NAT using UDP (although not impossible)
  - Setting up TCP or UDP *listeners* - i.e. servers - behind NAT is non-trivial (need to use port forwarding or DMZ)
  - And there aren't even enough IP address blocks to allocate each ISP customer *one* IPv4 address in some cases
  - What about mobile customers?
- IPv6 was intended to eliminate the need for NAT and vastly increase the IP address space, while fixing several other IPv4 shortcomings
- It has, so far, been a dismal failure

IPv6

- IPv6 uses 128 bits instead of 32; we now have 6×10^{23} addresses per square meter of the Earth's surface
- 340282366920938463463374607431768211456 addresses in total. This should be enough for now.
- IPv6 includes built-in security mechanisms, better multicast scalability, QoS features etc.
- Fragmentation is prohibited, greatly simplifying one of the major problems in IPv4

IPv6 Header Structure



The failure (so far) of IPv6

- IPv6 was introduced in December of 1998
- You'd think by now it was everywhere, and IPv4 was a historical memory... but no
- Some parts of the world have achieved a high level of IPv6 deployment (India is nearly 80% IPv6), while others are much lower (Australia is about 40%)
- CG-NAT and clever tricks with DNS (mainly the ubiquity of transparent content distribution networks) have eliminated many of the drivers for IPv6