

ECE 435 – Network Engineering

Lecture 13

Vince Weaver

<https://web.eece.maine.edu/~vweaver>

vincent.weaver@maine.edu

28 February 2023

Announcements

- Midterm on the 7th (week away)
- Project will be posted
- Interesting IPv4 address Allocation link:
<https://blog.daknob.net/mapping-44net/>
- Finding more available IP addresses proposal:
https://www.theregister.com/2022/06/01/ipv4_proposed_changes/



IPv4 Addresses

- Each IP address is 32-bits and has network address and host ID
- Can write many ways: decimal, hex, (all equivalent) but most common is dotted decimal (i.e. 12.34.56.78)
- Unique to *interface* not necessarily to *host*.
- Top level ran out in 2011, last NIC ran out 2019



Who Hands these Out?

- ICANN and various regional authorities Internet Corporation for Assigned Names and Numbers Internet Assigned Numbers Authority (IANA)
- Regional Internet Registrars
 - AfriNIC (Africa)
 - ARIN (N America),
 - APNIC (Asia-pacific)
 - LACNIC (latin america),
 - RIPE NCC (Europe and rest)



Subnets

- Number of hosts available can be larger than possible
- Divide network into subnets
- All hosts on subnet have the same prefix (left bits)
- Use subnet mask indicating the leftmost bits to use as subnet
- Can look like 255.255.255.0 meaning only bottom 8 bits are for host

Alternately can write this as 192.168.8.0/24 (24 is number of leading binary 1s in mask)



Classful IP Routing (No Longer Used)

- Routers just shifted right for A, B, and C class. Looked up A and B in table, C in hash table to find where to send
- Had a routing entry for each Class A (128), an entry for each class B (16k). Class C (2 million) a bit much, so hash table.
- Why so simple? In 80s memory and processors were expensive!
- Original classful addressing scheme (not used since 1993)



- Class A: 8 bit network (high bit 0) (24 bits of hosts) 0.0.0.0 to 127.255.255.255
- Class B: 16 bit network, (high bits 10) 128.0.0.0 to 191.255.255.255
- Class C: 24 bit network (high bits 110) 192.0.0.0 to 223.255.255.255
- Class D: multicast (high bits 1110) 224.0.0.0 to 239.255.255.255
- Class E: reserved (high bits 1111) 240.0.0.0 to 255.255.255.255



Reserved IP Ranges – Private Networks

- 10.0.0.0/8 private network (RFC 1918)
- 172.16.0.0/12 private network (RFC 1918)
- 192.168.0.0/16 Private Network (RFC 1918)



Reserved IP Ranges – Loopback

- 127.0.0.0/8 loopback (RFC 6890)



Reserved IP Ranges – Other

- 0.0.0.0/8 reserved for current network (RFC 6890)
- 100.64.0.0/10 shared address space (RFC 6598)
- 169.254.0.0/16 link-local (RFC 3927)
- 192.0.0.0/24 IETF (RFC 6890)
- 192.0.2.0/24 test (RFC 5737)
- 192.88.99.0/24 IPv6 to IPv4 relay (RFC 3068)
- 224.0.0.0/4 IP Multicast (class D) (RFC 5771)
- 240.0.0.0/4 Reserved (class E) (RFC 1700)
- 255.255.255.255/32 Broadcast (RFC 919)



Other IPv4 Conventions

- .0 represents a subnet
See <https://lwn.net/Articles/850374/> really old UNIX treated .0 (or all host bits 0) as another broadcast, there's a push to reclaim it as unicast
- .1 is often (but not always) a router
- it all host bits 1, broadcast for that subnet
- 255.255.255.255 is broadcast for device that doesn't know own IP yet (DHCP)
- What if /31, address 0 and 1?



Classless Inter-Domain Routing (CIDR)

- Running out (have run out) of network addresses
- For many groups, Class-A too big, Class-C too small (three bears problem?)
- Merge neighboring class-C together
- RFC 1519
- Scalability problem: each network takes up space in routing table
- Solution, group neighboring class Cs together
- With CIDR bit more complex.



- Triplet with IP address, subnet mask, outgoing line.
- In theory has to scan all. If multiple matches, one with longest mask is used.
- There are algorithms to make this go faster.



Local IP Routing

- If to same host, skip network.
- If on same subnet, send packet directly to destination (Ethernet)
- Otherwise, send to default router. See Linux route command. Often a “default router” 0.0.0.0/0. If doesn't match any other, sent out over default route
- If multiple network interfaces: If to this machine, deliver it, If to directly connected subnet, directly deliver, else deliver to next hop router



- How do we know if on network? If $((\text{hostIP XOR destip}) \& \text{subnetmask}) == 0$
- If local, how do we map IP to MAC? We'll see that in a minute.
- Due to CIDR, longest prefix matching. If match both a /21 and /24 then 24 is the one to send to as it's the longest.
- Data structures. Hashes? Trie?
 - Linux: two level hashing
 - BSD - trie (prefix tree)



Linux/UNIX routing setup

- Was `route` command, has been replaced by `ip route`
- `route add default gateway sets default gateway (router) for packets leaving the local network`
- also set up local subnets you are on, those packets don't need a router
- more complicated if you are configuring your Linux box to **be** a router



Linux/UNIX routing example

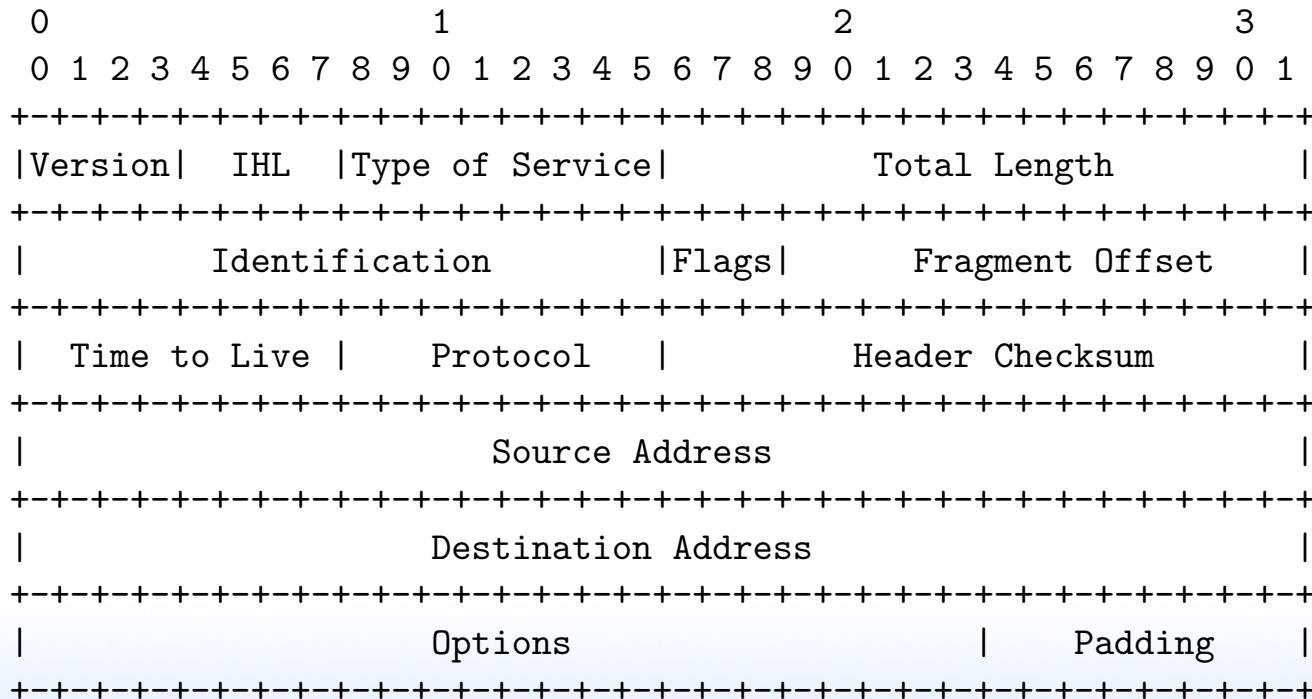
Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
default	192.168.8.2	0.0.0.0	UG	600	0	0	wlp2s0
link-local	0.0.0.0	255.255.0.0	U	1000	0	0	wlp2s0
192.168.8.0	0.0.0.0	255.255.255.0	U	600	0	0	wlp2s0



IPv4 Packet Format

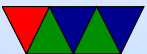
- Header, followed by data, multiple of 4-bytes, big-endian
- ASCII from RFC791 — <https://tools.ietf.org/html/rfc791>



- **Version** (4-bits) version number: IPv4 this is 4
- **Header Length** (4-bits) in 4-byte chunks: variable in size
Often is 5 (20 bytes) the minimum, max is 15 (60 bytes)
- **Precedence / Type of Service** (1 byte)
 - Precedence (RFC 791, high bits):
 - 111 (net control)
 - 110 (internetwork control)
 - 101 (critic/ecp)
 - 100 (Flash override)



- 011 (flash)
- 010 (intermediate)
- 001 (priority)
- 000 (routine)
- TOS (RFC 1349):
 - 1000 minimize delay
 - 0100 maximize throughput
 - 0010 maximize reliability
 - 0001 minimize cost
 - 0000 normal
 - 1111 maximize security



- R: reserved
- Replaced with DSCP (differentiated services code point) (RFC 2474) and ECN congestion (RFC 3168)
- **Total Length** (2 bytes) – max is 64kB
- **Identification** (2 bytes) – also called sequence, used in fragmentation
- **Fragmentation** (2 bytes) – fragmentation:
 - **flags** (3 bits): for fragmentation control.
 - high bit is always 0, (joke April Fools proposal ‘evil bit’)
 - next is “do not fragment”
 - last is “more fragments”



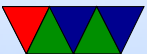
- **fragmentation offset** (13-bits): all but last fragment must be a multiple of 8-bytes as only have 13 bits to work with)
- **TTL** (1 byte) time-to-live, max routers allowed to pass through
 - (was supposed to be time, but ended up as a hop limit)
 - each router decreases TTL by one, if reaches zero discarded and ICMP error sent to source
 - Max is 255. why? prevent packets from wandering lost forever



- **Upper-layer protocol** (1 byte)
Originally in RFC 1700, now see www.iana.org
(ICMP=1, TCP=6, UDP=17) (many many more)
- **Header Checksum** (2 bytes)
 - Sum using 16-bit 1s complement, then complementing.
 - Not as strong as CRC-16, but faster and easier in software.
 - Only checksums header (not payload).
 - Must be recomputed each hop as TTL changes
- **Source address** (4 bytes)
- **Destination Address** (4 bytes)



- **Options** – not required. rare, debugging
 - security: how secret it is (usually ignored)
 - strict source: gives a list of IPs of routers to traverse
 - loose: list of routers not to miss
 - record route: record ips pass on way (debugging)
 - timestamp(debugging)
- Data



IPv4 Packet Fragmentation

- Ethernet MTU 1500 but IP MTU is 64k, so must break up larger packets
- Can be further broken up depending on MTU along way
- Final destination is responsible for reassembling
- Can mark packet “do not fragment”. What happens then if too big?
- All fragments have same ID/sequence number. Last fragment marked with 0 for “more fragments” flag. Position from fragmentation offset field



- Example: original, 3200 bytes of data
header id=x, more=1, offset=0, 1480 bytes
header id=x, more=1, offset=185 1480 bytes (x8?)
header id=x, more=0, offset=370 240 bytes
- Each fragment is a valid IP packet



Fragmentation Limits

- RFC 791 (1981)
- IPv4 Receivers must be able to handle fragmented packets with total re-assembled size of up to 576 bytes (modern OSes can generally handle up to 64k)
- IPv4 packets under 68 bytes can't be fragmented
- Picking the id/sequence number is complex see

<https://crnetpackets.com/2015/08/29/a-short-story-about-the-ip-id-field/>

essentially people wanted to re-use ID field for de-duplication but RFC 6864 says if DNF set you must



ignore ID

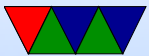


Problems with Fragments

- way to notify other side of missing fragments
- last fragment is usually short (wasting resources)
- receiver must hold in RAM fragments to be reassembled.
- can DoS by sending lots of fragments but none complete
- fragments do not have TCP or UDP header so firewall can't easily filter
- Most modern implementations will always disable fragmentation on TCP connections (setting DNF) and instead rely on path-mtu-discovery



- <https://blog.cloudflare.com/ip-fragmentation-is-broken/>



Path MTU Discovery

- Automatically determine the MTU (max transmission unit) between hosts
- Originally for routers, now also for endpoints
- Process
 - Set DNF bit on packets
 - Any router where packet size too big drops packet and sends back error via ICMP
 - Source reduces MTU and tries again until it gets through



Path MTU Issues

- If MTU gets smaller, will get notice and can adjust. No way to easily find if MTU gets bigger
- Problems if ICMP blocked
 - Complete 3-way handshake can happen (small packets) but then drop all actual traffic. “black hole connection”
 - Various workarounds for this. Force MTU to be Ethernet everywhere? Use TCP to probe size, treat packet drops as MTU issue not congestion?



Security Issues with Fragments

- ICMP/UDP larger than MTU, cannot be reassembled
- TCP “Teardrop” attack, send fragments with overlapping offsets, confuse/crash machines



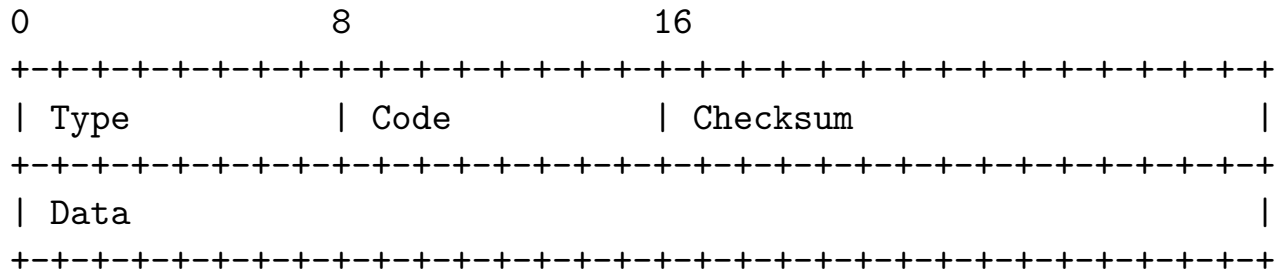
Errors

- What happens when something goes wrong with your packet?
- Does a router just drop it?
- Or does it try to let the sender know?



ICMP

- Internet Control Message Protocol
- Carried as a payload in an IP packet
- IP header type 1
- Some sysadmins block ICMP. Why?



A Selection of ICMP Types/Codes

- DESTINATION UNREACHABLE, Also if MTU is too small but do-not-fragment set
- SOURCE QUENCH – should slow transmission rate (congestion), This is now usually done in transport layer
- REDIRECT – try the other router path
- TIME EXCEEDED – exceeded TTL, traceroute uses this
- PARAMETER PROBLEM – illegal value in header
- ECHO, ECHO_REPLY – see if machine is up
- TIMESTAMP, TIMESTAMP_REPLY – performance debug



ping

- Mike Muuss in 1983
`http://ftp.arl.army.mil/~mike/ping.html`
- Like sonar ping (Hunt for Red October), not any of the backronyms you might find.
- Ping the duck
- ICMP ECHO packet, waits for ECHO reply. Prints timing info, etc.
- Used to just say “host is alive”. People would make machines called elvis.



Malicious pings – Ping of Death

- Ping of death – crash any machine on network (late 90s)
 - Technically not a ping bug, but fragmentation
 - Ping typically 56 bytes, but can be 64k
 - Technically not valid, but most will try anyway
 - 64k ping broken into 8 fragments
 - Maximum can specify is 65528, add in 20 for header, 65548
 - This is bigger than 65536, buffer overflow on reassemble



Malicious pings – Other

- Ping flood – could be used as DoS
- Broadcast ping to x.x.x.255 (no longer works)



traceroute

- Van Jacobson in 1987 (also wrote tcpdump)
- Uses ICMP
- *not* tracer-t
- Send packet with $TTL=1$, when sends ICMP error message know where first hop is
- Send packet with $TTL=2$, find next
- Linux traceroute sends UDP packets as originally ICMP requests weren't supposed to generate ICMP errors
- Sends 3 packets, lists all 3 results



Dynamic Host Configuration Protocol (DHCP)

- RFC2131
- To get on network need IP, subnet mask, default router
- Can we automatically get this?

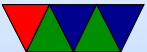


DHCP Protocol

- Device broadcasts, asking for address
- Server can respond with a fixed one (setup in config file) or handle out dynamically from range
- To avoid need for server on each subnet, can pass through
- Details
 - Broadcast DHCPDISCOVER on UDP port 67.
 - All servers send DHCPOFFER on port 68
 - Send DHCPREQUEST, respond with DHCPACK



- Timer, needs to re-request before timer is out or server might give to someone else
- Get a “lease” from the server. Why short vs long lease?
- Can see this all inaction with `dhclient -v`



Setting up DHCP server

- Static vs Dynamic (how hand out static addresses?)
- Be careful to not hand out on network you don't own
- Recent Linux systemd DNS debate (whether to fall back to default DNS router if can't get specified one)
- Network booting
 - PXE
 - bootp / tftp

