Networking 6: IP

i.g.batten@bham.ac.uk

Contents

- IP as a concept
- IPv4 addressing
- IPv6 addressing
- Packets and routing

Why IP?

- Far and away the dominant networking protocol of the past thirty years
- A single network layer, over which all transports can run, and…
- …a single network layer which can run over all available lower layers
- “inter” net — between networks

of different characteristics

What is IP?

- A unreliable, unsequenced datagram service
- You put an address on a packet and the network makes an attempt (“best efforts”) to deliver it somewhere, hopefully the right place.
- There is no checksum covering the data, so even protection from corruption is the responsibility of upper layers
- Hard to imagine a network layer which offers less

Why did IP win?

- Easy to implement, and implemented on all “Computer Science Favourites” of the early 1980s (Multics, TOPS-10, TOPS-20, Lisp Machines, Berkeley Unix)
  - Berkeley Unix, in turn, became the operating system of choice for Unix workstations, the dominant computer science environment of the late 80s and early to mid 1990s.
- Works over everything, from long-haul radio to exotic high-speed fibre.
- Has/had the Support of US DoD, (D)ARPA and NSF
- Actually rather good as well

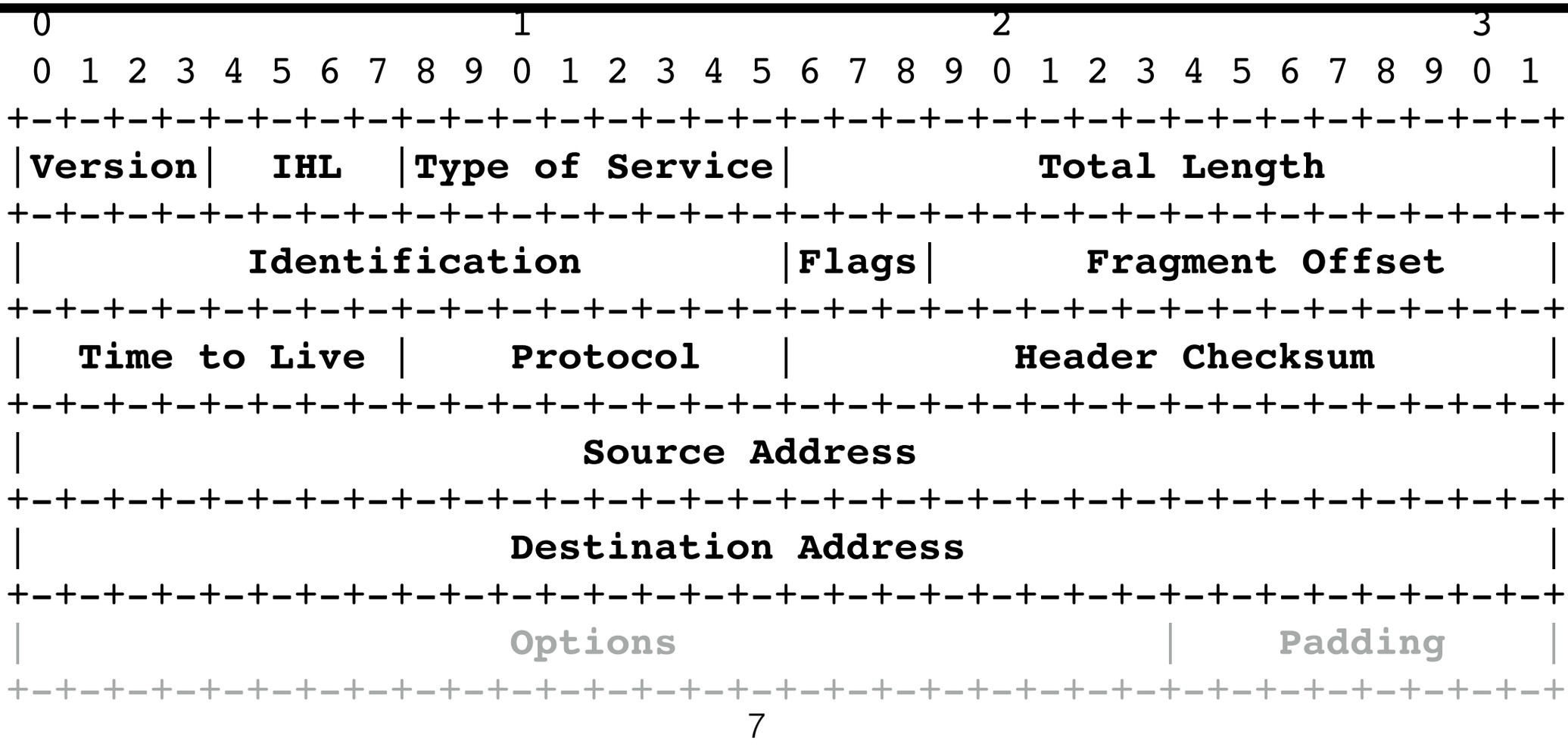
History

- Proposed in 1974 paper [1]
- Experimental versions 0 to 3 described in Internet Experimental Notes (IENs 2, 26, 28, 41, 44, 54)
- IPv4 described in RFC760, January 1981.
- Updated by RFC791, September 1981, which is still current (1349, 6864 and 2472 describe and clarify some little-used extensions).
- **You should read RFC791**

[1] Vinton G. Cerf, Robert E. Kahn, "A Protocol for Packet Network Intercommunication", IEEE Transactions on Communications, Vol. 22, No. 5, May 1974 pp. 637–648

Packet Format

- Each row is 32 bits, four bytes, 1 **word**. Options are optional, so a typical header is 20 bytes (5 words)
- Note fields aren't byte aligned but are 16-bit aligned: this is a 1970s "word" design with a nod to pdp11s.



ToS: Slight change today

- The 8 bits of “Type of Service” are now split up.
  - Six bits (8–13 inclusive) of Differentiated Services Code Point, which is used to do prioritisation for VoIP and other time-sensitive traffic
  - Two bits (14 and 15) of Explicit Congestion Notification, which is used to control overload on some networks
- Paradoxically, networks modern enough to support these are usually well engineered enough not to need them, while the people that need them don’t support them.

IHL

- IHL is the length of the header in 32 bit words.
- It is usually 5 (20 bytes)
- If >5 it means there are options present
  - IP options are mostly obsolete, useless, insecure or some combination.
  - Routine for firewalls to strip all options, on general principles: no reason not to do this.

Time to Live

- How many more seconds or hops the packet is valid for
- Decremented on each passage through a router (or every second, but that rarely happens)
- Initial value sets maximum diameter of the Internet: recommended value is 64, but there are various alternatives (255 in some versions of Solaris, down to 30 in some old versions of Windows).
- Generates ICMP Time Exc

Protocol Numbers

- 1 is ICMP: simple “code, details” error messages
- 6 is TCP: streams of data, which we will talk about next
- 17 is UDP: a thin datagram layer over IP
- https://en.wikipedia.org/wiki/List_of_IP_protocol_numbers is an menagerie of everything else: routing, experimental and obsolete.

ICMP Errors

- Time (Hop Count) Exceeded (usually a loop)
- Destination Unreachable (wide variety of reasons)
- Source Quench (“shut up I’m busy!”)
- Echo Request / Reply (Ping)
- Look up the rest

Total Length

- You might naively think this is measured in the same units as the header length, but no! It's measured in bytes.
- This makes sense: word-aligning the header is sensible, even today, but padding every packet to a multiple of 4 bytes would be hard word, even today.

Header Checksum

- NOT a secure hash: you can change one byte in the header and recompute the checksum based solely on the old checksum, the old value of the byte and new values of the byte.
- This is necessary in order to make updating the TTL field efficient
- Guards against accidental corruption, and nothing else.

Flags and Fragments

- IP has support for splitting packets into fragments so that a large packet (say, 1500 bytes) can travel over a network which only supports smaller packets (say, the old 576 byte minimum limit). The maximum packet size on a network is called the MTU (media, sometimes maximum, transmission unit).
- This is inefficient: the packets are only reassembled by the receiver, and loss of one fragment requires end-to-end retransmission of the whole packet
- Fragmentation is deprecated, and many firewall simply drop fragments. There are protocols designed to avoid it, and IPv6 explicitly does not support it (and has a much larger minimum MTU, *de jure* 1280, *de facto* 1560).
- Dropping fragments breaks some protocols (IKE, particularly) which rely on it: the usual and correct response is “tough: fix your protocol” (and there is a fix for IKE now).
- I will briefly explain it, because sadly it is still necessary in some scenarios

Flags

- Bit 0: reserved, must be zero (but see RFC3514 “Evil Bit”)
- Bit 1: Don’t Fragment - if the packet won’t fit through a link, drop it
- Bit 2: More fragments

Fragmentation

- If you have a packet too large for a link, put the IP header and as much data as possible into the first fragment. Add a unique Identification field and set the MF (“more fragments”) bit to 1.
- For all fragments but the last, use the same header, same ID, MF=1, but set the “fragment offset” to be the offset in bytes in the original packet of this fragment’s payload.
- For the last fragment, the same, but MF=0.

Example

- Original packet: 20 byte header, 900 bytes of payload, total size 920 bytes.
- Arrives at network which can only deal with 400 byte packets
- Need to split into three fragments of size 400, 400, 160, total 960 bytes (900 bytes of payload, plus three IPv4 headers of 20 bytes each).

Example

	Original Packet	Fragment 1	Fragment 2	Fragment 3
MF	0	**1**	**1**	**0**
ID	0	**1234**	**1234**	**1234**
Offset	0	**0**	**380**	**760**
Total Length	920	**400**	**400**	**160**
(Payload)	(900)	**(380)**	**(380)**	**(140)**

Reassembly

- Fragments are **NOT** reassembled when they pass from a network with a small MTU to one with a large MTU - routers don't have the RAM to buffer, and the fragments might travel by different routes anyway
- Fragments are **ONLY** reassembled by the **destination system**.
- Fragments are buffered until the destination has the complete set, then processed after reassembly.
  - DoS attack: just send a constant stream of fake fragments which never form a complete packet.
  - Reassembly timers not well defined
  - Firewalls today often just drop anything with MF=1 or Offset>0.
  - IPv6 has no concept of fragmentation, which is a good thing.

Path MTU Discovery

- Fix today: discover the minimum MTU along the route packets will follow.
- Send a packet with DF=1 (do not fragment).
- If you get a “Fragmentation Needed” error, reduce the size of packets to that destination (algorithms vary)
- Periodically re-probe, in case the route has changed and you can increase.
- Naive firewalls drop all ICMP: if you drop Fragmentation Needed ICMP messages then bad stuff happens.

32 bit addresses

- At the time, insanely large. The Arpanet reached a peak of 113 nodes by 1983, then split in half by the Arpanet / MILNet separation.
  - The question is not “how could they be so short-sighted as not to use 48 or 64 bits?” Rather “isn’t it amazing they didn’t use 16 or 24 bits?”
- Original concept of IP was to have 8 bits indicating the site, and 24 bits to specify a machine at that site.
  - Only proposed users were big US universities and companies, US government and a small number of defence-related operations, all in NATO.
- This was seen as wrong very quickly: 256 sites simply not enough, even in the early 1980s.

Notation

- Conventionally written as four decimal numbers, each encoding one byte (wastefully), separated by dots. Printable form 7..15 bytes long (4 times 1..3 digits plus three dots).
- Typically not zero-padded (usually 192.168.1.1 rather than 192.168.001.001), but sometimes people do use %03d.
- Hexadecimal would have been much better (you can encode, decode and mask by eye) but hexadecimal wasn't used much in the 1970s. We're lucky it wasn't octal, which was! IPv6 is hex, as we will see.

1001 0011	1011 1100	1100 0000	1111 1010
147	188	192	250
9 3	b c	c 0	f a

Routing Decisions

- All IP addresses identify a network (the leftmost part of the address) and the host on that network (the rest of the address). We will define “leftmost part” and “rest” soon.
- A router ignores the host on network part for all non-local addresses, looks up the network in a routing table, and chooses which interface to send the packet out through.
  - “Default network” handles all other cases.
- A router close to the centre of the inter-network needs to know about most networks.
- And at the time IP was designed, 64 KILO bytes was a LOT of RAM (maximum address space on a pdp11).

Priority: performance on limited hardware, 1980-style

- With the speeds and feeds of the era, building large distributed routing tables was hard. It was essential to keep the number of networks about which information needed to be exchanged to a minimum.
- Limiting it to 256 networks was unreasonable, but there had to be a low limit.
- 32 bits = ~ 4 billion addresses, in a era where a few thousand computers would be seen as an upper bound. Efficient allocation did not matter and was not a design goal: they anticipated utilisation of a fraction of a percent.
- The priority was being able to make routing decisions quickly on realistic hardware (roughly, a DEC LSI 11/23 “fuzzball”: 64kB per process maximum). Those decisions are with us still today.

“Classful” Addresses

- If the address starts with a 0, the first 8 bits identify the network, the remaining 24 bits identify the host on that network. 1.x.y.z through to 127.x.y.z
- Gives 128 “Class A” addresses to large sites that get 2^{24} (~16 million) hosts each. These went to MIT, BBN, Ford, DEC, Boeing, the UK government, etc.
- HP now has net 15 and DEC’s network 16, via its purchase of Compaq who had bought DEC.
- Net 0 not used, net 127 reserved for loopback (almost exclusively 127.0.0.1). Instantly wastes over 16 million address (2^{25}-1).

“Classful” Addresses

- If the address starts with 10, the first 16 bit identify the network, the remaining 16 bits the host on that network. 128.x.y.z through to 191.x.y.z
- Gives 2^{14} (16384) “Class B” addresses with 2^{16} (65536) hosts each.
- Initially easy to get for smaller universities and companies
  - Birmingham had **two**, one applied for centrally, one applied for to use in CS by Bob Hendley and me, not used after 1990, now sold (we didn’t see a penny of it).

“Classful” Addresses

- If the first three bits are 110, the first 24 bits identify the network, the remaining eight the host on the network. 192.x.y.z through 223.x.y.z
- Gives 2^{21} (2097152) “Class C” addresses with 2^{8} (256) hosts per network.

“Classful” Addresses

- Remaining space used for multicast (“Class D”, first four bits 1110, 224.x.y.z through 239.x.y.z) and reserved for experimental use (“Class E”, 1111, 240.x.y.z through 255.x.y.z).
- Multicast never really achieved much outside private networks, so address space that was allocated is wildly excessive.

Classes

	0..7	8..15	16..23	24..31
Class A	Net	Host		
Class B	Net		Host	
Class C	Net			Host

Wasteful Allocation

- Total reachable space is 87% of the available addresses
- But no university, not even MIT, has 16 million hosts, so Class A space very sparsely occupied
- Very few universities or companies have 65536 hosts, so Class B space very sparsely occupied
  - In both cases, even fewer hosts externally accessible
- Class C space was initially available, in bulk, to anyone with an Internet connection (Fulcrum Communications as it then was had 18 Class Cs, 4608 addresses, for <500 employees). This is not untypical.
- Estimates vary, but it's unlikely that today more than 25% of address space is usefully deployed. Huge shortages outside USA, Canada and western Europe (for practical purposes, Cold War era NATO).

But easy for routers

- Most of the early Internet was in fact the holders of the Class A addresses.
- Routers first looked at the address to see if it was “local”: is the destination directly connected to the router via some sort of network connection? If so, send the packet direct to that destination.
- Otherwise, they looked at the first bit of an address. If it was zero, they looked up the first byte in a 128-entry table of “next hops”: the IP numbers of other routers that are directly connected (LAN, WAN) and are believed to be “closer” to the destination.
    - With 32 bit addresses, such a 128 entry table occupies 512 bytes. Easy, even in 1980.
    - The original ARPAnet backbone was “net 10”, later re-purposed as we will see.
- If it’s found, send the packet to that router.
- Otherwise look up remaining two or three bytes in a more complex table (some sort of tree or hash table).
- Otherwise use the default route, if it exists.

Routing vs Memory

- Today you can have a fully populated routing table for each /24 in 128MB (2^{27}) of RAM (ie, a Raspberry Pi or an iPhone can function as a core router)
  - Insert rant about non-breaking punctuation here.
- There is no current need to do this, but you could have a unique destination for every IP number (all 32 bits) in 32GB of RAM (2^{35}) (a reasonably spec'd desktop PC, a small server)
- Routing now not a big computational problem: you simply index into a sparsely populated array rather than using complex trees and hash tables
- Hardly worth even caching the last eight (or whatever) destinations (a common trick of the past): processor will do it for itself if it's worth doing.

Sub-Netting

- People used the large amount of address space to plan their internal networking by structuring the “host” part of the network.
- Users of a Class B could treat their 2^{16} addresses as 256 Class C networks each of 256 hosts (which is what Birmingham do).
- Users of a Class A could treat their 2^{24} addresses as 65536 networks each of 256 hosts, or (with a lot of care and complexity) 256 groups each of 256 networks each of 256 hosts
- Practical limits of the time meant that you didn’t want more than ~100 hosts on an Ethernet anyway.
- By late 1980s, you could “subnet” on non-byte boundaries, too
  - Systems that can’t subnet, or can’t subject on arbitrary boundaries, are now obsolete; the hacks used to work around it are of historical interest only (and vile) — look up “Proxy ARP” if you have a strong stomach.
- Outside world sees one network, internally everyone knows the extra information about the layout of addresses

Netmasks

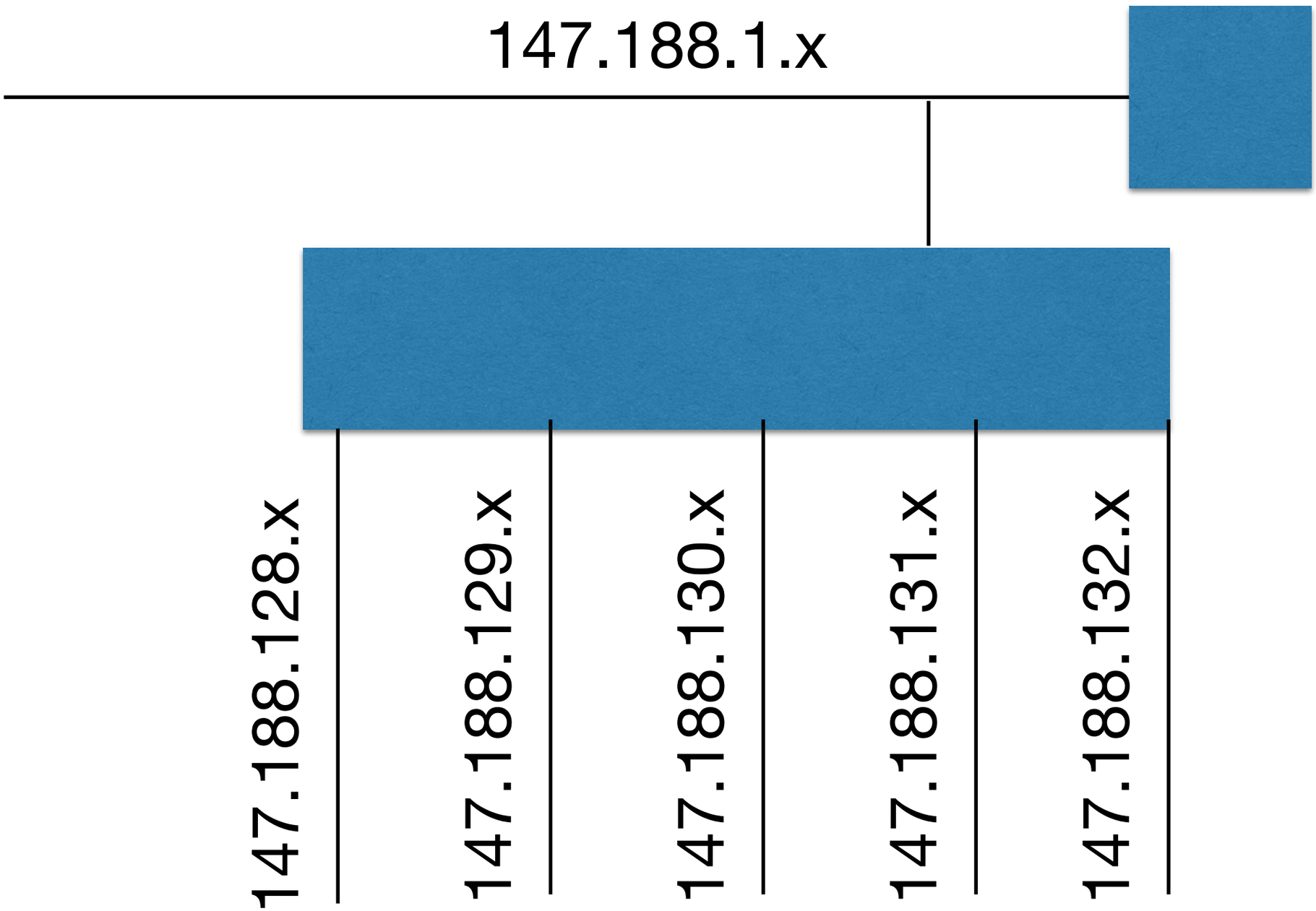
- Originally notated as a netmask: the bit pattern which can be logical-and'd with an address to yield a network number.
- Class A (later /8, as we will see) is 255.0.0.0,
  - 255 = 11111111. 10.1.2.3 & 255.0.0.0 = 10.0.0.0.
- Class B (/16) is 255.255.0.0
- Class C (/24) is 255.255.255.0
- A Class C used as 32 networks each of 8 hosts is 255.255.255.248 (/29): 248 = 11111000.

Subnets

	0..7	8..15	16..23	24..31
Class A	Net	Host		
Class B	Net		Host	
Class C	Net			Host
B, 255.255.255.0	Net		**Subnet**	Host
A, 255.255.255.0	Net	**Subnet**		Host
C, 255.255.255.248	Net			5 bit subnet

3 bit host

Sub-netting


147.188.1.x
147.188.128.x
147.188.129.x
147.188.130.x
147.188.131.x
147.188.132.x

CIDR and Slash Notation

- Problems of waste with Classful networking and need for more flexible sub-netting combined to produce Classless Interdomain Routing, CIDR.
- Every network address has a “netmask” which describes how much of it is network and how much of it is host.
- Class A is now a “slash eight” (MIT is 18.0.0.0/8), Class B is now a “slash sixteen” (Bham is 147.188.0.0/16) and Class C is a “slash twenty four” (FTEL is, amongst others, 192.65.220.0/24).

And in the other direction…

- To make routing tables more compact, a group of eight contiguous Class Cs can be placed together under a /21, or sixteen contiguous Class Bs under a /12.
- This is called “super netting”, but is now less useful as routing table size is not an issue.

Non-Byte Masks

- This extends trivially to boundaries which are not classful.
- So an ISP wanting to give prosumers some IP numbers can hand out a /28, which gives the user 16 IP numbers.
  - My home network is 81.187.150.208/28, giving me 81.187.150.208 through to 81.187.150.223, 14 hosts and two different broadcast addresses.

Recovering the Class As

- Some of the Class As were recovered, by a variety of threats, cajoling and bankruptcy (and Stanford being cool and stand-up and giving it back voluntarily in exchange for a bunch of Class Bs)
- Widespread allocation had stopped at Net 57 anyway (Societe Internationale de Telecommunications Aeronautiques S.C.R.L.)
- Remainder and returned networks were broken up and issued in varying size allocations
- http://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml

RFC1918

- Class A Network 10 became free when the Arpanet backbone was closed down.
- So 10.0.0.0/8, along with the available 172.16.0.0/12 (172.16.x.y through to 172.31.x.y) and 192.168.0.0/16 (192.168.x.y) were allocated for private use. These **must not** be routed outside private domains.
- This usually requires “address translation”, which we will cover later.

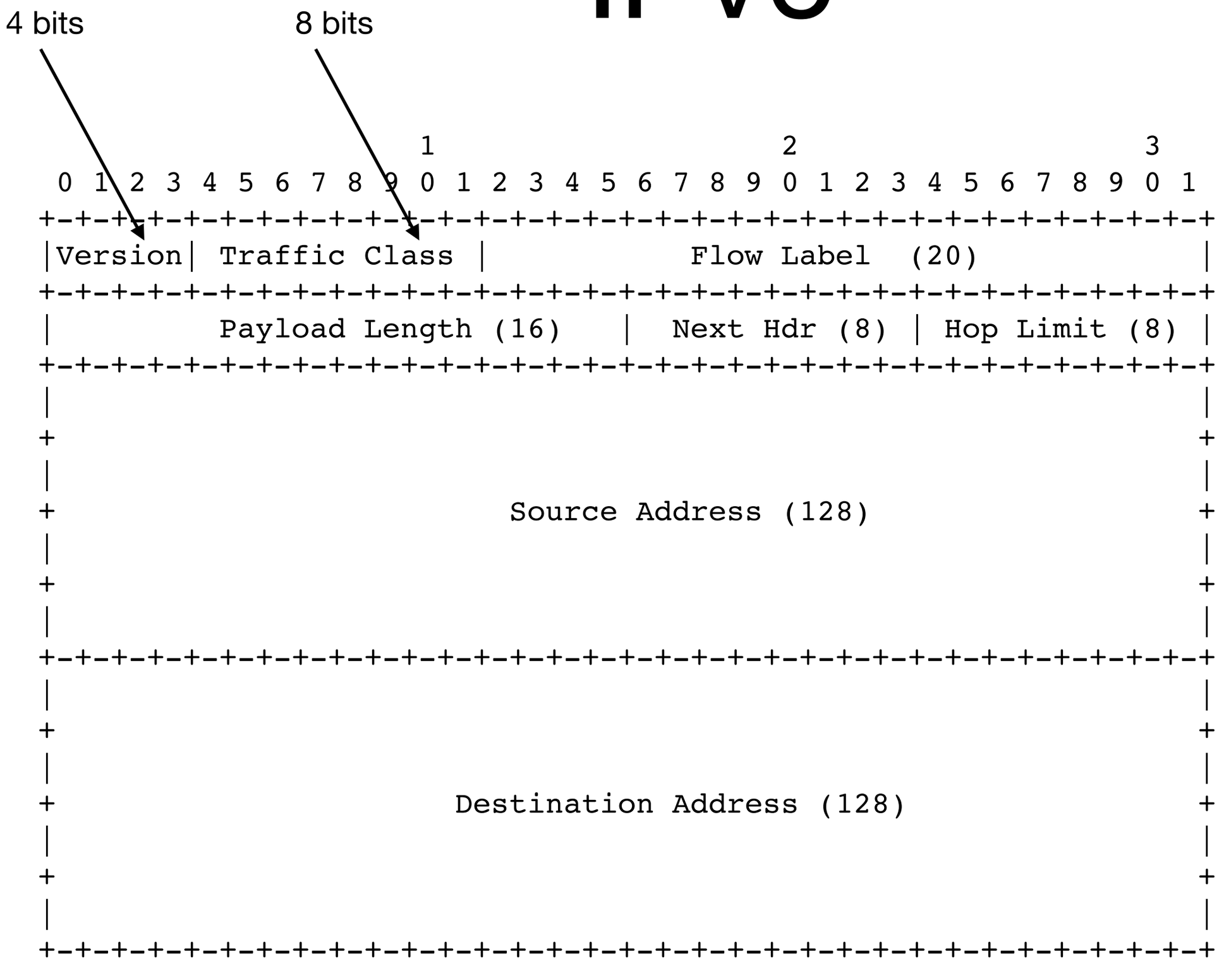
IPv6: IP with big addresses

- 32 bit addresses have run out (last /8s allocated in February 2011).
- 128 bit addresses gives 2^{96} times more addresses (ie, 2^{96} IPv6 addresses for each IPv4 address).
- Population of planet is ~2^{33} (8 billion)
- 2^{33} people, 2^{128} addresses, 2^{95} addresses per person.
- 2^{95} = 39614081257132168796771975168 (~ 4 x 10^{28})
    - That's quite a lot of computers, virtual machines and stuff. Each.

In reality, more like 2^{64}

- Minimum allocation unit is a /64; even your mobile phone will probably get a /64. IPv6 is “really” a 64 bit protocol.
- Intention is that with 64 bits available after the routed “prefix”, allocation of addresses on local networks is much easier
- 2^{64} still gives 2^{31} prefixes each (~2bn per person).
    - In fact, only 2^{61} addresses available to allocate, so 2^{28} each (~240 million per person)
- These are prefixes: each prefix is a network of up to 2^{32} devices!

IPv6


4 bits
8 bits
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Version
Traffic Class
Flow Label (20)
Payload Length (16)
Next Hdr (8)
Hop Limit (8)
Source Address (128)
Destination Address (128)

Header is subset of IPv4, plus one new field

- Version = 6
- Traffic Class: same as IPv4, 6 bits for DiffSrv, 2 bits for ECN.
- Flow Label: for labelling related flows of data, and the subject of much debate as to how and why it should work. Idea is that routers can keep related flows on the same path to reduce jitter and reordering.
- Payload length: length of remaining data
- Next Header: **type, not size**, of the next header (equivalent to IPv4 Protocol field, and uses same values)
- Hop Limit: as TTL in IPv4

Flow Classifiers

- Idea is that you have 20 bits to mark “related” traffic, so that routers can process it in a way which minimises reordering, delay variation, etc.
- And can do this without needing to figure out the flow from the TCP, UDP, etc header that follows.
- Much debate about correct use, and not obvious that just looking at TCP headers is not better.
- Rare example of IPv6 engaging in research speculation, rather than just simplifying IPv4.

IPv6 Addresses

- Standard format is a hex string, broken into 16 bit chunks with colons, leading zeros suppressed
  - 2001:8b0:129f:a90f:60c:ceff:fedd:f68
- “::” means “as many zeros as fit here”
  - 2001:8b0:129f:a90f:: = 2001:8b0:129f:a90f:0:0:0:0

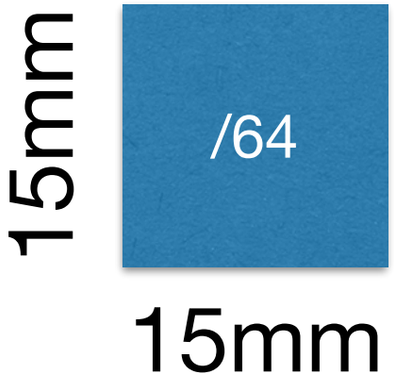
IPv6 is 64 bits

- IPv6 is “really” 64 bits, not 128 bits, in that no allocations should be done which are smaller than a /64 and routing tables do not process any network smaller than a /64. All allocations must be multiples of 4 (/64, /60, /56, etc).
- The idea is that every house, phone, office has at least a /64, and that gives 64 bits in which addresses of a small number of machines can be allocated either randomly or by simple mechanisms.
- Allocations /96s and /112s happen inside data centres, that is a private matter for consenting customers.
- Typical home allocation is a /56 or /48: 256 or 65536 prefixes.
- Typical business allocation is a /48, /44 or /40.

IPv6 reservations

- ::/128 “uninitialised”, ::1/128 “loopback” (note only one address, not 16 million!)
- ::ffff/96 IPv4 mapping (ie ::ffff:1.2.3.4)
- fc00::/7 Private address space
  - fc00::/8 is unmanaged
  - fd00::/8 is divided into /48 prefixes: generate 40 random bits for your site. See RFC4193.
- fe80::/8 link local,
- 2001:db8::/32 documentation etc (and a few others for similar purposes)
- 2000::/3 allocated as single block for normal use.
  - 2^{61} address blocks for end-users.

2^{61} is a LOT



- Do not listen to people complaining this is “wasteful”.
- Do not listen to people who say “ah, they said 2^{32} was a lot”.
- 2^{61} is 2^{28} /64 prefixes for every single human being currently alive.
- 2^{61} is a /64 prefix for every 2.2 cm^2 of the planet.
- 2^{61} is 135 000 /64 prefixes per watt of energy we currently produce, or 13 /64 prefixes per watt of solar energy that strikes the earth.
  - ie, if we could convert all solar energy 100% efficiently, and covered the entire planet with these 100% efficient solar cells, we would have a power budget of ~75mW per /64 prefix available.
- Every 2.2 square centimetres of the planet can have its own /64.
- Meanwhile, 2^{32} is <1 per human being which is a serious problem.
- **We do not need to use the bottom 64 bits anything other than locally.**

Consequences

- We can assign a separate IPv6 address to each service running on a computer, to make it easier to move service A to another machine without moving service B.
- We can assign a separate global IPv6 address to every device in a building, so an end to RFC1918-style local addresses.
- Yes, IPv6 has local addresses. Do not do this. Don’t let middle-aged men tell you the addresses are “scarce” (see above) or this is more “secure” (it probably isn’t).

IPv6 Routing Tables

- Will in the long-term require the complexity originally used for IPv4
- At the moment, somewhat sparse (sadly)
- 1TB is 2^{40} bytes, so 2^{61} is 2^{21} TB, which isn't going to happen any time soon. But nor is 2^{64} allocated networks.

IP in operation

- Sender: choose an interface we believe to be closer to the destination, and send the packet
- Recipient: if the packet is for us, process locally.
  - Otherwise, send it on.
- We will cover routing in more detail later, so we are just trying to get the general flavour here

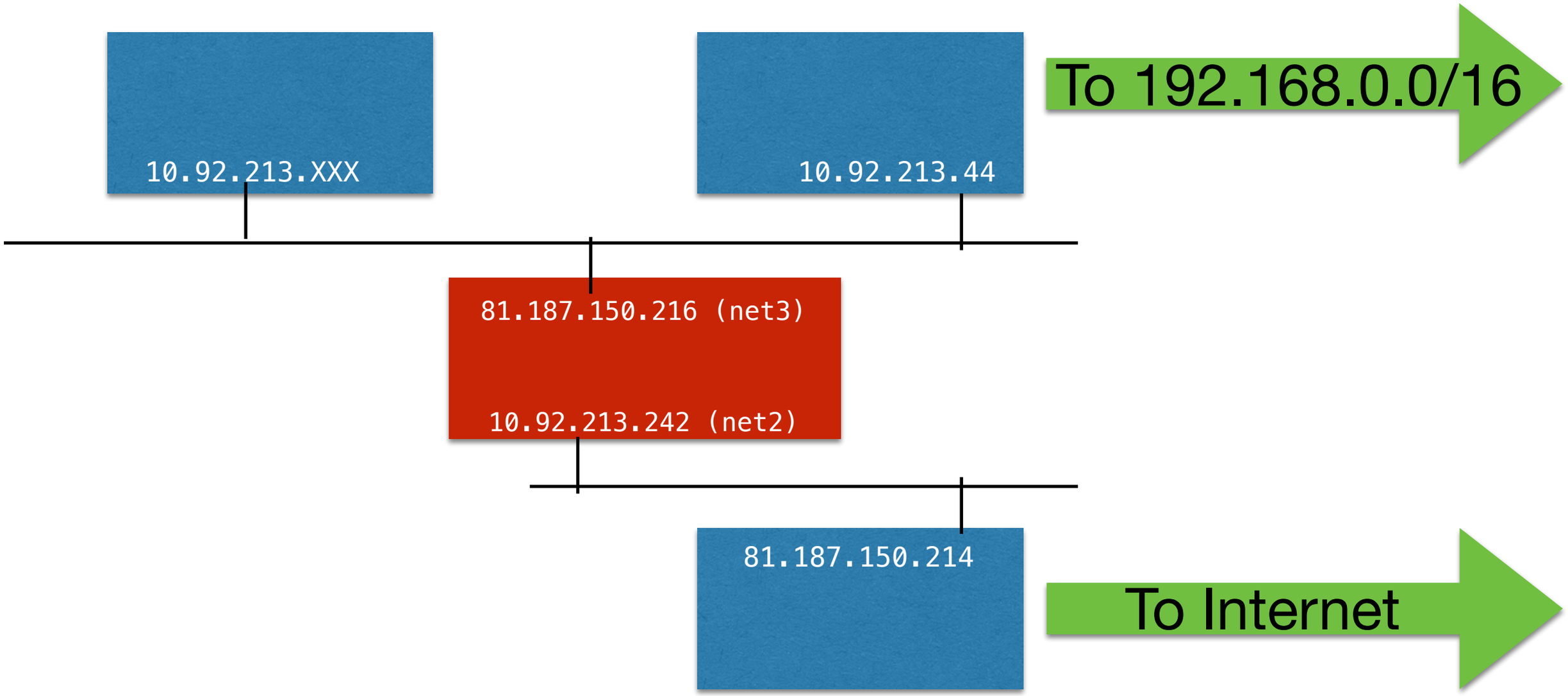
Simple Example

```
igb@ossec-sol:~$ netstat -nrv

IRE Table: IPv4
  Destination             Mask           Gateway           Device  MTU  Ref Flg  Out  In/Fwd
------------------- --------------- ------------------- ------ ----- --- --- ----- ------
default               0.0.0.0         81.187.150.214                0   2 UG     8426       0
10.92.213.0           255.255.255.0   10.92.213.242       net2    1500   6 U    757664       0
81.187.150.208        255.255.255.240 81.187.150.216      net3    1500   5 U      1854       0
127.0.0.1             255.255.255.255 127.0.0.1           lo0     8232   2 UH        0       0
192.168.0.0           255.255.0.0     10.92.213.44                  0   1 UG        0       0
```

Machine with two network interfaces
Note net2 is RFC1918, probably “internal”, and net3 is not, probably “external”
192.168/16 and default are additional routes

In Pictures


10.92.213.XXX
10.92.213.44
To 192.168.0.0/16
81.187.150.216 (net3)
10.92.213.242 (net2)
81.187.150.214
To Internet

Routing Decisions

- Decrement the TTL (and do so each time you've held on to the packet for a second, not that that happens).
- Look at each destination we know about, starting with the longest mask and working to the shortest
- Here we have locally connected ethernets (net2 and net 3)
- Traffic to 10.92.213.0/24 and 81.187.150.208.28 is "local" and goes direct over ethernet
- Traffic to 192.168.0.0 is sent over the ethernet to 10.92.213.44 (a **gateway**)
- Traffic to anywhere else is sent over the ethernet to 81.187.150.214 (again, a gateway)
- Machine might itself be a router: whether it forwards packets that arrive on one interface with addresses on the other side is a policy decision.

IP on Ethernet

- On an ethernet, or other “point to multi-point” network, how do we find the MAC address of the next hop?
- IPv4 uses ARP: Address Resolution Protocol
  - Simple, old and frighteningly insecure
  - Ask “WHO HAS” a particular IP number
  - Station with that IP number, or someone who claims to know about it, tells us. Ripe for exploitation, as we will learn in network security lectures
- IPv6 uses “Neighbo(u)r Discovery Protocol” to do similar job, with ICMPv6 messages 135 (solicitation) and 136 (advertisement)

IP on point-to-point links

- If a link is point to point, you just send the packet down it.
- Might be a physical point-to-point link (a serial line of some sort, perhaps) or might be a tunnel (packets encapsulated in other packets). We will talk about tunnels later.
- “The internet is a large open field, with many tunnels running underneath it”.

IP to Gateways

- When sending a packet to a gateway, you look up the gateway address using ARP if necessary, but the IP destination remains the ultimate destination. Gateways don’t appear in the IP header.
- Gateways at other end of point to point links just involve sending the packet.

Hop Counts

- Each packet has a Time To Live (TTL).
- Decremented each time the packet is processed, or whenever a router holds on to it for more than one second (rare today).
- When it hits zero, packet is discarded and an error report (“ICMP Time Exceeded”) is generated.
- Typical initial value 30–64, depending on current estimates of “diameter” of internet.
- Prevents packets circulating endlessly in case of routing loops or similar.
- Requires re-computation of the header checksum in IPv4
  - Fast algorithms used to recompute it based on knowing what has changed: this isn’t a secure hash
- IPv6 doesn’t have a header checksum, relying instead on lower layers being reliable and upper layers being sensible
  - Good reason for routers and switches to have ECC RAM. 1 flipped bit in 10^{12} = 1 silently broken packet per second at 1Tb/sec.

IP

- Basic, best efforts, unsequenced, unreliable
- Packets can be corrupted, dropped, delayed, reordered
- Fragmentation possible but not recommended
- ICMP