15-441 Computer Networking Lecture 14 – IP Wrap Up

Peter Steenkiste
Departments of Computer Science and
Electrical and Computer Engineering

15-441 Networking, Spring 2008
http://www.cs.cmu.edu/~dga/15-441/S08

Outline

- The recurring IP address space problem
- IPv6.
- NAT.
- Tunneling / Overlays
- Network Management
  - » Autoconfiguration
  - » SNMP (notes only)

IP Address space

- Address space crunch 1: Classful routing
  - » 128 "class A" blocks of 2^24 addresses (too big)
  - » 16k "class B" blocks of 2^16 addresses (still too big)
  - » 2M "class C" blocks (often too small)
  - » Result: Exceptionally wasteful allocation
    - – MIT still has 18.0.0.0/8 -- 16M addresses for 30k people
  - » Solution: CIDR ("cider"). Classful Inter-Domain Routing.
    - – Removed classness.
    - – Now can route on arbitrary power of two boundary
    - – "slash" notation: /8 = 255.0.0.0, /16 = 255.255.0.0, etc.

IP address space 2

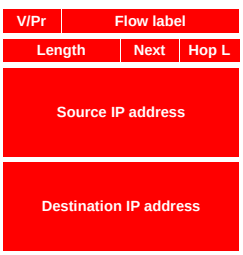
- How many IP addresses? 4B
- How many…
  - » People? 6.5B
  - » Cell phones? 2.2B (roughly)
  - » Embedded computers? (???, but huge)
  - » What happens if you network all of the devices in your house?
  - » Big reality in 2005: We're about 50% used.
    - – CIDR
    - – Tighter allocation policies; voluntary IP reclamation
    - – NAT (later today)
- Will it happen? Maybe, maybe not. But I wouldn't bet against it.
- But even if not, put on architecture hat and think about the problems…

Efficiency vs. Convenience

- Exact allocation vs. Routing Table Size
  - » If I have 20 computers, how do you alloc?
    - – /27 = 32 (30 usable): Waste 12
    - – /28 = 16 (14) + /29 (8, 6 usable): Waste 4
    - – Can't get any better…
    - – We've traded a bit of address efficiency for two BGP routing table entries.
- Exact allocation vs. Future Growth
  - » Suppose you buy more computers
    - – Could add a new netblock (more table entries)
    - – Could move to a bigger one (re-addressing)
      - Update computers
      - Update routers
      - Update DNS
      - Update address allocation registries
      - Maybe have to fix some hard-coded addresses, if you were bad
  - » Partial solution: DHCP (we'll talk about later today)

IP v6

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



IPv6 Addressing

- 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.

010	Registry	Provider	Subscriber	Sub Net	Host

IPv6 Cleanup – Consider Router Architectures

- Common case: Switched in silicon ("fast path")
- Weird cases: Handed to CPU ("slow path", or "process switched")
- Typical division:
  - Fast path: Almost everything
  - Slow path:
    - 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 can't (really) use things on the slow path for data flow
    - If it became popular, they'd be in the soup!
- Other speed issue: Touching data is expensive. Designers like to minimize accesses to packet during forwarding

IPv6 Header Cleanup: Checksum

- No checksum
- Why checksum just the IP header?
  - Efficiency: If packet corrupted at hop 1, don't waste b/w transmitting on hops 2..N.
  - Useful when corruption frequent, b/w expensive
  - Today: Corruption rare, b/w cheap

IPv6 Header Cleanup: Option Handling

- IPv4 options: Variable length header field. 32 different options.
  - 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
    - Things like "source route" implemented only at intermediate hops

IPv6 Fragmentation Cleanup

- IPv4:

- IPv6:
  - Discard packets, send ICMP "Packet Too Big"
    - Similar to IPv4 "Don't Fragment" bit handling
  - Sender must support Path MTU discovery
    - Receive "Packet too Big" messages and send smaller packets
  - Increased minimum packet size
    - Link must support 1280 bytes;
    - 1500 bytes if link supports variable sizes
- 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 gradual deployment.
- Two complementary mechanisms:
  - dual stack operation: IP v6 nodes support both address types
  - tunneling: tunnel IP v6 packets through IP v4 clouds
- Alternative is to create IPv6 islands, e.g. corporate networks, ...
  - Use of form of NAT to connect to the outside world
  - NAT must not only translate addresses but also translate between IPv4 and IPv6 protocols

IPv6 Discussion

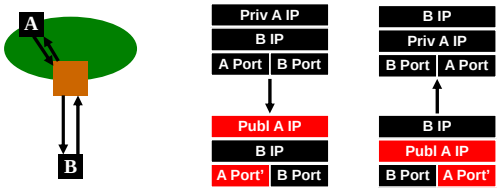
- IPv4 Infrastructure got better
  - » Address efficiency
  - » Co-opted IPv6 ideas: IPSec, diffserv, autoconfiguration via DHCP, etc.
- Massive challenge
  - » Huge installed base of IPv4-speaking devices
  - » Chicken & Egg problem
    - – Who's the first person to go IPv6-only?
- Steady progress in deployment.
  - » Most hosts & big routers support.
  - » Long-term: The little devices will probably force IPv6
    - – Used now on many mobile phones in Japan

NATs and Tunnels

- NATs originally invented as a way to help migrate to a hybrid IPv4 IPv6 world
  - » Took on a life of their own
  - » May have substantially delayed IPv6 deployment by reducing address pressure!
  - » You probably encounter them every day
- Tunnels: Coming up after NATs.

Network Address Translation

- NAT maps (private source IP, source port) onto (public source IP, unique source port)
  - » reverse mapping on the way back
  - » destination host does not know that this process is happening
- Very simple working solution.
  - » NAT functionality fits well with firewalls



Types of NATs

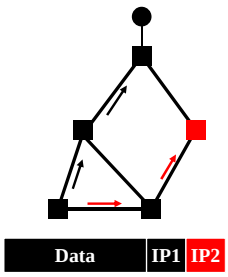
- Bi-directional NAT: 1 to 1 mapping between internal and external addresses.
  - » E.g., 128.237.0.0/16 -> 10.12.0.0/16
  - » External hosts can directly contact internal hosts
  - » Why use?
    - – Flexibility. Change providers, don't change internal addrs.
    - – Need as many external addresses as you have *hosts* - can use sparse address space internally.
- "Traditional" NAT: Unidirectional
  - » Basic NAT: Pool of external addresses
    - – Translate source IP address (+checksum,etc) only
  - » Network Address Port Translation (NAPT): What most of us use
    - – Also translate ports.
      - E.g., map (10.0.0.5 port 5555 -> 18.31.0.114 port 22) to (128.237.233.137 port 5931 -> 18.31.0.114 port 22)
    - – Lets you share a single IP address among multiple computers

NAT Considerations

- NAT has to be consistent during a session.
  - » Set up mapping at the beginning of a session and maintain it during the session
    - – Recall 2nd level goal 1 of Internet: Continue despite loss of networks or gateways
    - – What happens if your NAT reboots?
  - » Recycle the mapping that the end of the session
    - – May be hard to detect
- NAT only works for certain applications.
  - » Some applications (e.g. ftp) pass IP information in payload
  - » Need application level gateways to do a matching translation
  - » Breaks a lot of applications.
    - – Example: Let's look at FTP
- NAT is loved and hated
  - - Breaks many apps (FTP)
  - - Inhibits deployment of new applications like p2p (but so do firewalls!)
  - + Little NAT boxes make home networking simple.
  - + Saves addresses. Makes allocation simple.

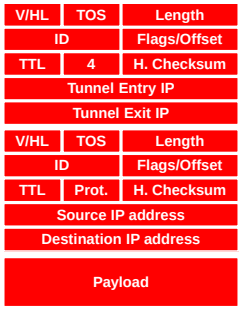
Tunneling

- Force a packet to go to a specific point in the network.
  - » Path taken is different from the regular routing
- 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 using IP source routing option
- 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.



Tunneling Example

Tunneling Considerations

- Performance.
  - » Tunneling adds (of course) processing overhead
  - » Tunneling increases the packet length, which may cause fragmentation
    - – BIG hit in performance in most systems
    - – Tunneling in effect reduces the MTU of the path, but end-points often do not know this
- Security issues.
  - » Should verify both inner and outer header
  - » E.g., one-time flaw: send an ip-in-ip packet to a host. Inner packet claimed to come from "trusted" host. Bypass firewalls.

Tunneling Applications

- Virtual private networks.
  - » Connect subnets of a corporation using IP tunnels
  - » Often combined with IP Sec
    - – (Amusing note: IPSec itself an IPv6 spinoff that was backported into IPv4)
- Support for new or unusual protocols.
  - » Routers that support the protocols use tunnels to "bypass" routers that do not support it
  - » E.g. multicast
- Force packets to follow non-standard routes.
  - » Routing is based on outer-header
  - » E.g. mobile IP

Overlay Networks

- A network "on top of the network".
  - » E.g., initial Internet deployment
    - – Internet routers connected via phone lines
      - An overlay on the phone network
  - » Tunnels between nodes on a current network
- Examples:
  - » The IPv6 "6bone", the multicast "Mbone" ("multicast backbone").
- But not limited to IP-layer protocols…
  - » Can do some pretty cool stuff:

Overlay Networks 2

- Application-layer Overlays
  - » Application Layer multicast (last week)
    - – Transmit data stream to multiple recipients
  - » Peer-to-Peer networks
    - – Route queries (Gnutella search for "briney spars")
    - – Route answers (Bittorrent, etc. -- project 2)
  - » Anonymizing overlays
    - – Route data through lots of peers to hide source
      - (google for "Tor" "anonymous")
  - » Improved routing (Resilient Overlay Networks)
    - – (Shameless plug of my own research)
    - – Detect and route around failures *faster* than the underlying network does.
- Overlays provide a way to build interesting services / ideas without changing the (huge, hard to change) IP infrastructure.
- Design Q: When are overlays good?
  - » Functionality between small(er) group of people w/out requiring global state/changes/etc.

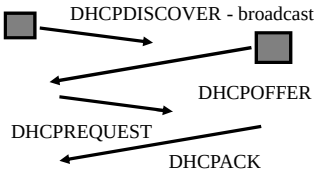
Network Management

- Two sub-issues:
  - Configuration management
    - How do I deal with all of these hosts?!
  - Network monitoring
    - What the heck is going on on those links?
    - (Left for notes, not talking about)

Autoconfiguration

- Recall other problem with address space: It's a pain to re-address
  - Affects allocation size, ease of switching ISPs, etc.
- IP address, netmask, gateway, hostname, etc., etc.
  - Typing by hand: Ugh!
- IPv4 option 1: RARP (Reverse ARP)
  - Data-link protocol
  - Uses ARP format. New opcodes: "Request reverse", "reply reverse"
  - Send query: Request-reverse [ether addr], server responds with IP
- IPv4 option 2: DHCP
  - Dynamic Host Configuration Protocol
  - ARP is fine for assigning an IP, but is very limited
  - DHCP can provide the kitchen sink

DHCP



- DHCPOFFER
  - IP addressing information
  - Boot file/server information (for network booting)
  - DNS name servers
  - Lots of other stuff - protocol is extensible; half of the options reserved for local site definition and use.

DHCP Features

- Lease-based assignment
  - Clients can renew. Servers really should preserve this information across client & server reboots.
- Provide *host* configuration information
  - Not just IP address stuff.
  - NTP servers, IP config, link layer config,
  - X window font server (wow)
- Use:
  - Generic config for desktops/dialin/etc.
    - Assign IP address/etc., from pool
  - Specific config for particular machines
    - Central configuration management

IPv6 Autoconfiguration

- Serverless ("Stateless"). No manual config at all.
  - Only configures addressing items, NOT other host things
    - If you want that, use DHCP.
- Link-local address
  - 1111 1110 10 :: 64 bit interface ID (usually from Ethernet addr)
    - (fe80::/64 prefix)
  - Uniqueness test ("anyone using this address?")
  - Router contact (solicit, or wait for announcement)
    - Contains globally unique prefix
    - Usually: Concatenate this prefix with local ID -> globally unique IPv6 ID
- DHCP took some of the wind out of this, but nice for "zero-conf" (many OSes now do this for both v4 and v6)

Network "Management"

- Management is still not too well defined
- Understanding network status, responding intelligently, etc
- Managing configurations
  - How do you "program" the network?

Management: Monitoring

- What to do when there is a problem?
  - » Loss of connectivity, complaints of slow throughput, ..
- How do you know how busy your network is?
  - » Where are the bottlenecks, is it time for an upgrade, redirect traffic, ..
- How can you spot unusual activity?
  - » Somebody attacking a subnet, ..
- These are all hard problems that are typically addressed using multiple tools, but the ability to monitor network status is a common requirement.
  - » "Static" information: what is connected to what?
  - » Dynamic information: what is the throughput on that link?

Common Monitoring Tools

- SNMP
  - » Simple Network Management Protocol
    - – Device status
      - 5 minute traffic average on outbound links
      - Amount of disk space used on server
      - Number of users logged in to modem bank
      - Etc.
    - – Device alerts
      - Line 5 just went down!
  - » Netflow
    - – Detailed traffic monitoring
      - Break down by protocol/source/etc.
      - ("Who's serving 5 terabytes of briney spars photos??")

Simple Network Management Protocol (SNMP)

- Protocol that allows clients to read and write management information on network elements.
  - » Routers, switches, …
  - » Network element is represented by an SNMP agent
- Information is stored in a management information base (MIB).
  - » Have to standardize the naming, format, and interpretation of each item of information
  - » Ongoing activity: MIB entries have to be defined as new technologies are introduced
- Different methods of interaction supported.
  - » Query response interaction: SNMP agent answers questions
  - » traps: agent notifies registered clients of events
- Need security: authentication and encryption.

MIB

- Information is represented in an object tree.
  - » To identify information you specify a path to a leaf
  - » Can extend MIB by adding subtrees
  - » Different standard bodies can expand different subtrees
    - – E.g. Ethernet and ATM groups are independent
- Uses ASN.1 standard for data representation.
  - » Existing standard
  - » How is information stored?
  - » How is information encoded on the wire (transfer syntax)