Connecting

G54ACC – IP and Up

Lecture 5

Contents

- Address shortages
- Middleboxes
- Naming
- Quality of Service

Contents

- Address shortages
  - NAT
  - Layer violation
  - ICE, STUN
- Middleboxes
- Naming
- Quality of Service

Address Shortages

- IPv4 supports 32 bit addresses
  - 95% allocated already (300,000 netblocks)
  - June 2011 (global), Feb 2012 (regional) zero-day
  - ...yet #connected devices is exploding
- IPv6 supports 128 bit addresses
  - So not a problem?
  - ...except for the routing protocols
  - ...and all the associated services needing to move

Network Address Translation

- Private Addressing, RFC1918
  - 172.16/12, 192.168/16, 10/8
  - Should never be externally routed
- Traditional NAT, RFC3022; see also RFC2663
  - Use private addresses internally
  - Map into a (small) set of routable addresses
  - Use source ports to distinguish connections
  - Requires IP, TCP/UDP header rewriting
    - Addresses, ports, checksums at least
- Not a security mechanism!

Layer Violation

- Information leaking from one layer to another
  - Generally considered poor form
  - Sometimes useful for features or performance
- NAT often causes this to explode
  - Commonly where addresses are (ab)used as host identifiers
    - They're not, they're addresses for routing to interfaces
    - E.g., FTP, SIP; often anything where subsequent connections need to be setup
  - Also IP fragmentation, some options, some ICMP
  - And both ends may be NATted

ICE, STUN

- Session Traversal Utilities for NAT, RFC5389
  - Client attempts to characterise NAT behaviour using a third-party server “on the outside”
- Interactive Connectivity Establishment, RFC5245
  - Commonly used with SIP, SDP (for voice-on-IP)
    - In general, “offer/answer” protocols
  - Uses STUN (or TURN or ...) to determine and select from set of “candidate transport addresses”
  - Selected addresses are then propagated and used

Contents

- Address shortages
- Middleboxes
  - The end-to-end argument
- Naming
- Quality of Service

Middleboxes

- A NAT is an example of a (transparent) middlebox
- There are others
  - Firewalls
  - Proxies
  - Caches
- They often provide very useful services
- But can be a complete pain
  - Buggy, unreliable
  - Incomplete protocol support (ICMP, &c.)

End-to-end Argument

- Salzer *et al*. "End-to-End Arguments in System Design". ACM Transactions on Computer Systems, 2(4), pages 277-288, 1984.
  - Earlier version in 1981
- Functions whose implementation requires application involvement should not be provided at lower layers
  - Unless partial implementation helps performance
- Will revisit this later in course

Contents

- Address shortages
- Middleboxes
- Naming
  - Name service
  - DNS protocol
- Quality of Service

Naming

- IP addresses are all very well but
  - Not especially human-readable
  - Not always appropriate granularity
- HOSTS.TXT
  - A file (/etc/hosts) mapping names-numbers
  - Originally transferred to all hosts using FTP
  - Simple, but not terribly automatic or scalable
  - Scale via distributed hierarchical set of servers

DNS

- Domain Name Service, RFC1034/1035/2181
  - Client-Server protocol returning variety of records
  - Commonly uses UDP for queries but can use TCP
  - TCP used for bulk transfers between servers
- Hierarchy is “baked in”
  - Namespace divides into *zones*
  - Top Level Domains usually professionally managed
  - Root servers know how to get everywhere
- Not a 1:1 mapping between names and numbers!
  - E.g., Round-robin load-balancing

Name Service

- TLDs operated by registrars
- Delegate sub-domains to other registrars
  - ...and on down the hierarchy
- Eventually customer rents a subdomain/name
  - I.e., registrar installs appropriate records
- Setup primary and secondary servers
  - For subdomains
  - Separate IP netblocks, physical networks, &c
  - DNS is a *very* common single-point-of-failure

Queries

- Queries either *recursive* or *iterative*
  - A-B-C-D-A; or A-B-A, A-C-A, A-D-A
- Server either *authoritative* or *caching*
  - To discover authoritative requires query to root
  - Thus load on root servers is very high
- Caching server locally
  - Caches records each with an expiry time: *soft-state*
- Acquire zone's complete set via *zone transfer*
  - Often access controlled

Responses

- Name lookup uses following record types:
  - CNAME: name |-> canonical name
    - `www.cs.nott.ac.uk. 61272 IN CNAME pat.cs.nott.ac.uk.`
  - A: name |-> number
    - `pat.cs.nott.ac.uk.  68622  IN  A  128.243.20.9`
    - `pat.cs.nott.ac.uk.  68622  IN  A  128.243.21.19`
  - PTR: name (or number) |-> name
    - `9.20.243.128.in-addr.arpa.  39617 IN PTR pat.cs.nott.ac.uk.`
  - NS: domain |-> authoritative name server
    - `cs.nott.ac.uk.  10585  IN  NS  ns1.nottingham.ac.uk.`
    - `cs.nott.ac.uk.  10585  IN  NS  ns2.nottingham.ac.uk.`
    - `cs.nott.ac.uk.  10585  IN  NS  marian.cs.nott.ac.uk.`
    - `cs.nott.ac.uk.  10585  IN  NS  extdns1.warwick.ac.uk.`
    - `cs.nott.ac.uk.  10585  IN  NS  extdns2.warwick.ac.uk.`
  - MX: domain |-> mail exchange
    - `nott.ac.uk.  3600  IN  MX  1 mx191.emailfiltering.com.`
    - `nott.ac.uk.  3600  IN  MX  2 mx192.emailfiltering.com.`
    - `nott.ac.uk  3600  IN  MX  3 mx193.emailfiltering.com.`

Security

- DNS is quite insecure
  - Cache poisoning
    - Caching and soft-state mean bad data propagates and can persist for some time
    - Even if through a simple mistake
  - Man-in-the-middle attacks
    - Iterative/Recursive queries almost demand this
  - Name spoofing
    - How clear is *your* font?
    - How well can *your* users spell?

Contents

- Address shortages
- Middleboxes
- Naming
- Quality of Service
  - IntServ
  - DiffServ

Quality of Service

- What do you do when capacity < demand?
  - If capacity > demand, no need for QoS
  - Queuing (latency, jitter, loss) should be minimal
  - At least, in a stable network (cf. dynamic routing)
- Retrofitted to the Internet:
  - Integrated Services (IntServ)
  - Differentiated Services (DiffServ)
    - Cf. ATM where it was baked in from the start
- Neither are especially widely used
  - Inelastic *vs*. Elastic traffic: higher layer responses

IntServ *vs*. DiffServ

- IntServ
  - Operates on explicitly signalled *flows*
  - Flow setup specifies some QoS
  - Routers perform Connection Admission Control
- DiffServ
  - Operates on *traffic aggregates*
  - Label packets with desired service class
    - Low latency, low loss, high throughput defined
  - Routers apply queuing as operator sees fit

Summary

- Features are introduced to manage the changing environment
  - E.g., address shortages
- The law of *unintended consequences*
- The end-to-end argument is an *argument*!
- Dealing with NAT
- Naming and DNS
- Quality of service