CS 3700
Networks and Distributed Systems

Lecture 11: DNS + NAT
Outline

- DNS
- NAT
- Other middleboxes
Layer 8 (The Carbon-based nodes)

- If you want to...
  - Call someone, you need to ask for their phone number
    - You can’t just dial “P R O F M I S L O V E”
  - Mail someone, you need to get their address first
Layer 8 (The Carbon-based nodes)

- If you want to...
  - Call someone, you need to ask for their phone number
    - You can’t just dial “P R O F M I S L O V E”
  - Mail someone, you need to get their address first
- What about the Internet?
  - If you need to reach Google, you need their IP
  - Does anyone know Google’s IP?
Layer 8 (The Carbon-based nodes)

- If you want to...
  - Call someone, you need to ask for their phone number
    - You can’t just dial “P R O F M I S L O V E”
  - Mail someone, you need to get their address first

- What about the Internet?
  - If you need to reach Google, you need their IP
  - Does anyone know Google’s IP?

- Problem:
  - People can’t remember IP addresses
  - Need human readable names that map to IPs
Internet Names and Addresses

- Addresses, e.g. 129.10.117.100
  - Computer usable labels for machines
  - Conform to structure of the network
Internet Names and Addresses

- Addresses, e.g. 129.10.117.100
  - Computer usable labels for machines
  - Conform to structure of the network

- Names, e.g. www.northeastern.edu
  - Human usable labels for machines
  - Conform to organizational structure
Internet Names and Addresses

- Addresses, e.g. 129.10.117.100
  - Computer usable labels for machines
  - Conform to structure of the network
- Names, e.g. www.northeastern.edu
  - Human usable labels for machines
  - Conform to organizational structure
- How do you map from one to the other?
  - Domain Name System (DNS)
Before DNS, all mappings were in *hosts.txt*

- `/etc/hosts` on Linux
- `C:\Windows\System32\drivers\etc\hosts` on Windows
Before DNS, all mappings were in *hosts.txt*

- `/etc/hosts` on Linux
- `C:\Windows\System32\drivers\etc\hosts` on Windows

Centralized, manual system

- Changes were submitted to SRI via email
- Machines periodically FTP new copies of *hosts.txt*
- Administrators could pick names at their discretion
- Any name was allowed
  - `alans_server_at_neu_pwns_joo_lol_kthxbye`
Eventually, the *hosts.txt* system fell apart
Towards DNS

- Eventually, the *hosts.txt* system fell apart
  - Not scalable, SRI couldn’t handle the load
  - Hard to enforce uniqueness of names
    - e.g MIT
      - Massachusetts Institute of Technology?
      - Melbourne Institute of Technology?
  - Many machines had inaccurate copies of *hosts.txt*
Towards DNS

- Eventually, the *hosts.txt* system fell apart
  - Not scalable, SRI couldn’t handle the load
  - Hard to enforce uniqueness of names
    - e.g MIT
      - Massachusetts Institute of Technology?
      - Melbourne Institute of Technology?
  - Many machines had inaccurate copies of *hosts.txt*

- Thus, DNS was born
Outline

- DNS Basics
- DNS Security
DNS at a High-Level

- Domain Name System
- Distributed database
  - No centralization
- Simple client/server architecture
  - UDP port 53, some implementations also use TCP
  - Why?
- Hierarchical namespace
  - As opposed to original, flat namespace
  - e.g. .com → google.com → mail.google.com
Naming Hierarchy

- **Top Level Domains (TLDs)** are at the top
- **Maximum tree depth**: 128
- Each Domain Name is a subtree
  - .edu → neu.edu → ccs.neu.edu → [www.ccs.neu.edu](http://www.ccs.neu.edu)
- Name collisions are avoided
  - neu.com vs. neu.edu
Naming Hierarchy

- Top Level Domains (TLDs) are at the top
- Maximum tree depth: 128
- Each Domain Name is a subtree
  - .edu → neu.edu → ccs.neu.edu → www.ccs.neu.edu
- Name collisions are avoided
  - neu.com vs. neu.edu
Naming Hierarchy

- Top Level Domains (TLDs) are at the top
- Maximum tree depth: 128
- Each Domain Name is a subtree
  - .edu → neu.edu → ccs.neu.edu → www.ccs.neu.edu
- Name collisions are avoided
  - neu.com vs. neu.edu
Naming Hierarchy

- Top Level Domains (TLDs) are at the top
- Maximum tree depth: 128
- Each Domain Name is a subtree
  - .edu → neu.edu → ccs.neu.edu → www.ccs.neu.edu
- Name collisions are avoided
  - neu.com vs. neu.edu
Naming Hierarchy

- Top Level Domains (TLDs) are at the top
- Maximum tree depth: 128
- Each Domain Name is a subtree
  - .edu → neu.edu → ccs.neu.edu → www.ccs.neu.edu
- Name collisions are avoided
  - neu.com vs. neu.edu
Hierarchical Administration

- Tree is divided into zones
  - Each zone has an administrator
  - Responsible for the part of the hierarchy

- Example:
  - CCIS controls *.ccs.neu.edu
  - NEU controls *.neu.edu
Hierarchical Administration

- Tree is divided into zones
  - Each zone has an administrator
  - Responsible for the part of the hierarchy

Example:
- CCIS controls *.ccs.neu.edu
- NEU controls *.neu.edu
Hierarchical Administration

- Tree is divided into zones
  - Each zone has an administrator
  - Responsible for the part of the hierarchy

- Example:
  - CCIS controls *.ccs.neu.edu
  - NEU controls *.neu.edu
Server Hierarchy

- Functions of each DNS server:
  - Authority over a portion of the hierarchy
    - No need to store all DNS names
  - Store all the records for hosts/domains in its zone
    - May be replicated for robustness
  - Know the addresses of the root servers
    - Resolve queries for unknown names
Functions of each DNS server:
- Authority over a portion of the hierarchy
  - No need to store all DNS names
- Store all the records for hosts/domains in its zone
  - May be replicated for robustness
- Know the addresses of the root servers
  - Resolve queries for unknown names

Root servers know about all TLDs
- The buck stops at the root servers
Root Name Servers

- Responsible for the Root Zone File
  - Lists the TLDs and who controls them
  - ~272KB in size

```
com. 172800 IN NS a.gtld-servers.net.
com. 172800 IN NS b.gtld-servers.net.
com. 172800 IN NS c.gtld-servers.net.
```

- Administered by ICANN
  - 13 root servers, labeled A→M
  - 6 are anycasted, i.e. they are globally replicated
- Contacted when names cannot be resolved
  - In practice, most systems cache this information
Local Name Servers

- Each ISP/company has a local, default name server
- Often configured via DHCP
- Hosts begin DNS queries by contacting the local name server
- Frequently cache query results
Local Name Servers

- Each ISP/company has a local, default name server
- Often configured via DHCP
- Hosts begin DNS queries by contacting the local name server
- Frequently cache query results
Local Name Servers

- Each ISP/company has a local, default name server
- Often configured via DHCP
- Hosts begin DNS queries by contacting the local name server
- Frequently cache query results
Authoritative Name Servers

- Stores the name→IP mapping for a given host
Stores the name→IP mapping for a given host
Authoritative Name Servers

Where is www.neu.edu?

- Root
- edu
- neu

- Authority for ‘edu’
- Authority for ‘neu.edu’

Stores the name→IP mapping for a given host

Northeastern
Where is www.neu.edu?

- Stores the name ➔ IP mapping for a given host
Authoritative Name Servers

Where is www.neu.edu?

www.neu.edu = 155.33.17.68

Northeastern

Stores the name→IP mapping for a given host
Basic Domain Name Resolution

- Every host knows a local DNS server
  - Sends all queries to the local DNS server
Basic Domain Name Resolution

- Every host knows a local DNS server
  - Sends all queries to the local DNS server
- If the local DNS can answer the query, then you’re done
  1. Local server is also the authoritative server for that name
  2. Local server has cached the record for that name
Basic Domain Name Resolution

- Every host knows a local DNS server
  - Sends all queries to the local DNS server
- If the local DNS can answer the query, then you’re done
  1. Local server is also the authoritative server for that name
  2. Local server has cached the record for that name
- Otherwise, go down the hierarchy and search for the authoritative name server
  - Every local DNS server knows the root servers
  - Use cache to skip steps if possible
    - e.g. skip the root and go directly to .edu if the root file is cached
Recursive DNS Query

17

www.google.com

ns1.google.com

asgard.ccs.neu.edu

com

Root

ns1.google.com

com

www.google.com

ns1.google.com

asgard.ccs.neu.edu

com

Root
Recursive DNS Query

Where is www.google.com?
Recursive DNS Query

Where is www.google.com?
Recursive DNS Query

Where is www.google.com?

www.google.com

asgard.ccs.neu.edu

ns1.google.com

com

Root
Where is www.google.com?
Recursive DNS Query

Where is www.google.com?
Recursive DNS Query

Where is www.google.com?
Recursive DNS Query

Where is www.google.com?

- Puts the burden of resolution on the contacted name server

www.google.com
ns1.google.com
asgard.ccs.neu.edu
Root
com
Recursive DNS Query

- Puts the burden of resolution on the contacted name server
- How does asgard know who to forward responses too?
  - Random IDs embedded in DNS queries

Where is www.google.com?
Recursive DNS Query

- Puts the burden of resolution on the contacted name server
- How does asgard know who to forward responses too?
  - Random IDs embedded in DNS queries
- What have we said about keeping state in the network?

Where is www.google.com?
Iterated DNS query
Iterated DNS query

Where is www.google.com?

www.google.com

ns1.google.com

asgard.ccs.neu.edu

Root

com

ns1.google.com

asgard.ccs.neu.edu

Where is www.google.com?
Iterated DNS query

Where is www.google.com?

- www.google.com
- ns1.google.com
- com
- Root
- asgard.ccs.neu.edu
- com
- Root
- ns1.google.com
Iterated DNS query

- Contact server replies with the name of the next authority in the hierarchy

Where is www.google.com?

www.google.com

ns1.google.com

com

Root

asgard.ccs.neu.edu

Root

com
Iterated DNS query

- Contact server replies with the name of the next authority in the hierarchy

Where is www.google.com?
Iterated DNS query

- Contact server replies with the name of the next authority in the hierarchy

Where is www.google.com?
Where is www.google.com?

Contact server replies with the name of the next authority in the hierarchy

www.google.com

ns1.google.com

asgard.ccs.neu.edu

com

Root
Where is www.google.com?

- Contact server replies with the name of the next authority in the hierarchy

- ns1.google.com
- asgard.ccs.neu.edu
- Root
- com
Iterated DNS query

- Contact server replies with the name of the next authority in the hierarchy

Where is www.google.com?

ns1.google.com

www.google.com

asgard.ccs.neu.edu

Root

com
Iterated DNS query

Contact server replies with the name of the next authority in the hierarchy

“I don’t know this name, but this other server might”

This is how DNS works today

Where is www.google.com?
How many of you have purchased a domain name?

- Did you notice that it took ~72 hours for your name to become accessible?
- This delay is called DNS Propagation
How many of you have purchased a domain name?

Did you notice that it took ~72 hours for your name to become accessible?

This delay is called DNS Propagation

www.my-new-site.com
How many of you have purchased a domain name?

Did you notice that it took ~72 hours for your name to become accessible?

This delay is called DNS Propagation.
How many of you have purchased a domain name?

Did you notice that it took ~72 hours for your name to become accessible?

This delay is called DNS Propagation
DNS Propagation

- How many of you have purchased a domain name?
  - Did you notice that it took ~72 hours for your name to become accessible?
  - This delay is called DNS Propagation

- Why would this process fail for a new DNS name?
DNS Propagation delay is caused by caching

- Cached Root Zone File
- Cached .com Zone File
- Cached .net Zone File
- Etc.

asgard.ccs.neu.edu

Root

com

www.my-new-site.com

ns.godaddy.com
Caching vs. Freshness

- DNS Propagation delay is caused by caching

Where is www.my-new-site.com?

- Cached Root Zone File
- Cached .com Zone File
- Cached .net Zone File
- Etc.

www.my-new-site.com

ns.godaddy.com
DNS Propagation delay is caused by caching

- Cached Root Zone File
- Cached .com Zone File
- Cached .net Zone File
- Etc.

That name does not exist.

Caching vs. Freshness
Caching vs. Freshness

- DNS Propagation delay is caused by caching
  - Cached Root Zone File
  - Cached .com Zone File
  - Cached .net Zone File
  - Etc.

- Zone files may be cached for 1-72 hours
DNS Resource Records

- DNS queries have two fields: name and type
- Resource record is the response to a query
  - Four fields: (name, value, type, TTL)
  - There may be multiple records returned for one query
DNS Resource Records

- DNS queries have two fields: name and type
- Resource record is the response to a query
  - Four fields: (name, value, type, TTL)
  - There may be multiple records returned for one query
- What are do the name and value mean?
  - Depends on the type of query and response
DNS Types

- **Type = A / AAAA**
  - **Name = domain name**
  - **Value = IP address**
  - **A is IPv4, AAAA is IPv6**
DNS Types

- **Type = A / AAAA**
  - **Name = domain name**
  - **Value = IP address**
  - A is IPv4, AAAA is IPv6

**Query**

Name: [www.ccs.neu.edu](http://www.ccs.neu.edu)
Type: A
## DNS Types

- **Type = A / AAAA**
  - **Name = domain name**
  - **Value = IP address**
  - A is IPv4, AAAA is IPv6

<table>
<thead>
<tr>
<th>Query</th>
<th>Name: <a href="http://www.ccs.neu.edu">www.ccs.neu.edu</a></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Type: A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resp.</th>
<th>Name: <a href="http://www.ccs.neu.edu">www.ccs.neu.edu</a></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value: 129.10.116.81</td>
</tr>
</tbody>
</table>
DNS Types

- **Type = A / AAAA**
  - Name = domain name
  - Value = IP address
  - A is IPv4, AAAA is IPv6

- **Type = NS**
  - Name = partial domain
  - Value = name of DNS server for this domain
  - “Go send your query to this other server”

**Query**
- Name: [www.ccs.neu.edu](http://www.ccs.neu.edu)
- Type: A

**Resp.**
- Name: [www.ccs.neu.edu](http://www.ccs.neu.edu)
- Value: 129.10.116.81
DNS Types

- **Type = A / AAAA**
  - Name = domain name
  - Value = IP address
  - A is IPv4, AAAA is IPv6

- **Type = NS**
  - Name = partial domain
  - Value = name of DNS server for this domain
  - “Go send your query to this other server”

**Query**
Name: www.ccs.neu.edu
Type: A

**Resp.**
Name: www.ccs.neu.edu
Value: 129.10.116.81

**Query**
Name: ccs.neu.edu
Type: NS
DNS Types

- **Type = A / AAAA**
  - Name = domain name
  - Value = IP address
  - A is IPv4, AAAA is IPv6

- **Type = NS**
  - Name = partial domain
  - Value = name of DNS server for this domain
  - “Go send your query to this other server”

<table>
<thead>
<tr>
<th>Query</th>
<th>Name: <a href="http://www.ccs.neu.edu">www.ccs.neu.edu</a></th>
<th>Type: A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resp.</td>
<td>Name: <a href="http://www.ccs.neu.edu">www.ccs.neu.edu</a></td>
<td>Value: 129.10.116.81</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query</th>
<th>Name: ccs.neu.edu</th>
<th>Type: NS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resp.</td>
<td>Name: ccs.neu.edu</td>
<td>Value: 129.10.116.51</td>
</tr>
</tbody>
</table>
Type = CNAME
- Name = hostname
- Value = canonical hostname
- Useful for aliasing
- CDNs use this
DNS Types, Continued

- **Type = CNAME**
  - Name = hostname
  - Value = canonical hostname
  - Useful for aliasing
  - CDNs use this

**Query**
Name: `foo.mysite.com`
Type: CNAME
Type = CNAME
- Name = hostname
- Value = canonical hostname
- Useful for aliasing
- CDNs use this

Query
Name: foo.mysite.com
Type: CNAME

Resp.
Name: foo.mysite.com
Value: bar.mysite.com
DNS Types, Continued

- **Type = CNAME**
  - Name = hostname
  - Value = canonical hostname
  - Useful for aliasing
  - CDNs use this

- **Type = MX**
  - Name = domain in email address
  - Value = canonical name of mail server
DNS Types, Continued

- **Type = CNAME**
  - Name = hostname
  - Value = canonical hostname
  - Useful for aliasing
  - CDNs use this

- **Type = MX**
  - Name = domain in email address
  - Value = canonical name of mail server

Query
Name: foo.mysite.com
Type: CNAME

Resp.
Name: foo.mysite.com
Value: bar.mysite.com

Query
Name: ccs.neu.edu
Type: MX
DNS Types, Continued

- **Type = CNAME**
  - Name = hostname
  - Value = canonical hostname
  - Useful for aliasing
  - CDN’s use this

- **Type = MX**
  - Name = domain in email address
  - Value = canonical name of mail server
Reverse Lookups

- What about the IP→name mapping?
- Separate server hierarchy stores reverse mappings
  - Rooted at in-addr.arpa and ip6.arpa
- Additional DNS record type: PTR
  - Name = IP address
  - Value = domain name
- Not guaranteed to exist for all IPs
Reverse Lookups

- What about the IP→name mapping?
- Separate server hierarchy stores reverse mappings
  - Rooted at in-addr.arpa and ip6.arpa
- Additional DNS record type: PTR
  - Name = IP address
  - Value = domain name
- Not guaranteed to exist for all IPs

Query
Name: 129.10.116.51
Type: PTR
Reverse Lookups

- What about the IP→name mapping?
- Separate server hierarchy stores reverse mappings
  - Rooted at in-addr.arpa and ip6.arpa
- Additional DNS record type: PTR
  - Name = IP address
  - Value = domain name
- Not guaranteed to exist for all IPs

Query

Name: 129.10.116.51
Type: PTR

Resp.

Name: 129.10.116.51 Value: ccs.neu.edu
DNS as Indirection Service

- DNS gives us very powerful capabilities
  - Not only easier for humans to reference machines!
DNS as Indirection Service

- DNS gives us very powerful capabilities
  - Not only easier for humans to reference machines!

- Changing the IPs of machines becomes trivial
  - e.g. you want to move your web server to a new host
  - Just change the DNS record!
Aliasing and Load Balancing

One machine can have many aliases

www.reddit.com
www.foursquare.com
www.huffingtonpost.com
One machine can have many aliases

- www.reddit.com
- www.foursquare.com
- www.huffingtonpost.com
- david.choffnes.com
- alan.mislo.ve
- *.blogspot.com
Aliasing and Load Balancing

- One machine can have many aliases
  - www.reddit.com
  - www.foursquare.com
  - www.huffingtonpost.com
  - david.choffnes.com
  - alan.mislo.ve
  - *.blogspot.com

- One domain can map to multiple machines
  - www.google.com
  - Three servers connected to www.google.com
Content Delivery Networks
Content Delivery Networks
Content Delivery Networks
Content Delivery Networks
Content Delivery Networks
DNS responses may vary based on geography, ISP, etc
Outline

- DNS Basics
- DNS Security
The Importance of DNS

- Without DNS...
  - How could you get to any websites?
The Importance of DNS

- Without DNS...
  - How could you get to any websites?
- You are your mailserver
  - When you sign up for websites, you use your email address
  - What if someone hijacks the DNS for your mail server?
The Importance of DNS

- Without DNS...
  - How could you get to any websites?
- You are your mailserver
  - When you sign up for websites, you use your email address
  - What if someone hijacks the DNS for your mail server?
- DNS is the root of trust for the web
  - When a user types www.bankofamerica.com, they expect to be taken to their bank’s website
  - What if the DNS record is compromised?
Denial Of Service

- Flood DNS servers with requests until they fail
- October 2002: massive DDoS against the root name servers
  - What was the effect?
Denial Of Service

- Flood DNS servers with requests until they fail
- October 2002: massive DDoS against the root name servers
  - What was the effect?
  - ... users didn’t even notice
  - Root zone file is cached almost everywhere
Denial Of Service

- Flood DNS servers with requests until they fail
- October 2002: massive DDoS against the root name servers
  - What was the effect?
  - ... users didn’t even notice
  - Root zone file is cached almost everywhere
- More targeted attacks can be effective
  - Local DNS server → cannot access DNS
  - Authoritative server → cannot access domain
DNS Hijacking

- Infect their OS or browser with a virus/trojan
  - e.g. Many trojans change entries in /etc/hosts
  - *.bankofamerica.com → evilbank.com

- Man-in-the-middle

- Response Spoofing
  - Eavesdrop on requests
  - Outtrace the servers response
DNS Hijacking

- Infect their OS or browser with a virus/trojan
  - e.g. Many trojans change entries in /etc/hosts
  - *.bankofamerica.com → evilbank.com

- Man-in-the-middle

- Response Spoofing
  - Eavesdrop on requests
  - Outtrace the servers response
DNS Hijacking

- Infect their OS or browser with a virus/trojan
  - e.g. Many trojans change entries in /etc/hosts
  - *.bankofamerica.com $\rightarrow$ evilbank.com

- Man-in-the-middle

- Response Spoofing
  - Eavesdrop on requests
  - Outtrace the servers response
DNS Spoofing

dns.bofa.com

123.45.67.89
Where is bankofamerica.com?

123.45.67.89
DNS Spoofing

Where is bankofamerica.com?

123.45.67.89

123.45.67.89
DNS Spoofing

dns.bofa.com

dns.evil.com

Bank of America

123.45.67.89

66.66.66.93
DNS Spoofing

Where is bankofamerica.com?

123.45.67.89

Bank of America

66.66.66.93

dns.evil.com

dns.bofa.com

66.66.66.93
DNS Spoofing

Where is bankofamerica.com?

bankofamerica.com

66.66.66.93

dns.evil.com

66.66.66.93

123.45.67.89

bankofamerica.com

66.66.66.93
DNS Spoofing

How do you know that a given name → IP mapping is correct?

Where is bankofamerica.com?

66.66.66.93
DNS Cache Poisoning

dns.neu.edu

ns1.google.com
Where is www.google.com?
DNS Cache Poisoning

www.google.com = 74.125.131.26

dns.neu.edu

ns1.google.com
DNS Cache Poisoning

www.google.com = 74.125.131.26

bankofamerica.com = 66.66.66.92

dns.neu.edu

ns1.google.com
DNS Cache Poisoning

dns.neu.edu

ns1.google.com
DNS Cache Poisoning

Where is bankofamerica.com?
DNS Cache Poisoning

- Until the TTL expires, all queries for BofA to dns.neu.edu will return poisoned result
- Much worse than spoofing/man-in-the-middle
  - Whole ISPs can be impacted!
Solution: DNSSEC

- Cryptographically sign critical resource records
  - Resolver can verify the cryptographic signature

- Two new resource types
  - Type = DNSKEY
    - Name = Zone domain name
    - Value = Public key for the zone
  - Type = RRSIG
    - Name = (type, name) tuple, i.e. the query itself
    - Value = Cryptographic signature of the query results
Solution: DNSSEC

- Cryptographically sign critical resource records
  - Resolver can verify the cryptographic signature

- Two new resource types
  - Type = DNSKEY
    - Name = Zone domain name
    - Value = Public key for the zone
  - Type = RRSIG
    - Name = (type, name) tuple, i.e. the query itself
    - Value = Cryptographic signature of the query results

Creates a hierarchy of trust within each zone
Solution: DNSSEC

- Cryptographically sign critical resource records
  - Resolver can verify the cryptographic signature
- Two new resource types
  - Type = DNSKEY
    - Name = Zone domain name
    - Value = Public key for the zone
  - Type = RRSIG
    - Name = (type, name) tuple, i.e. the query itself
    - Value = Cryptographic signature of the query results

Prevents hijacking and spoofing
Solution: DNSSEC

- Cryptographically sign critical resource records
  - Resolver can verify the cryptographic signature
- Two new resource types
  - Type = DNSKEY
    - Name = Zone domain name
    - Value = Public key for the zone
  - Type = RRSIG
    - Name = (type, name) tuple, i.e. the query itself
    - Value = Cryptographic signature of the query results
- Deployment
  - On the roots since July 2010
  - Verisign enabled it on .com and .net in January 2011
  - Comcast is the first major ISP to support it (January 2012)
DNSSEC Hierarchy of Trust

Root Zone (ICANN)

.com (Verisign)

dns.bofa.com
DNSSEC Hierarchy of Trust

Where is bankofamerica.com?

Root Zone (ICANN)

.com (Verisign)

IP: 123.45.67.89
Key: < >
SIG: x9fnskflkalk

dns.bofa.com
DNSSEC Hierarchy of Trust

Root Zone (ICANN)

.com (Verisign)

Where is bankofamerica.com?

dns.bofa.com

IP: 123.45.67.89
Key: <
SIG: x9fnskflkalk
DNSSEC Hierarchy of Trust

Where is bankofamerica.com?

IP: 123.45.77.89
Key: < >
SIG: x9fnskflkalk

dns.bofa.com

Root Zone (ICANN)

.com (Verisign)
DNSSEC Hierarchy of Trust

Where is bankofamerica.com?

Root Zone (ICANN)
.com (Verisign)

IP: 66.66.66.93
Key: < >
SIG: 9na8x7040a3
DNSSEC Hierarchy of Trust

Where is bankofamerica.com?

Root Zone (ICANN)

.com (Verisign)

IP: 66.66.66.93
Key: < >
SIG: 9na8x7040a3

dns.evil.com
September 2003: Verisign created DNS wildcards for *.com and *.net
- Essentially, catch-all records for unknown domains
- Pointed to a search website run by Verisign
- Search website was full of advertisements
September 2003: Verisign created DNS wildcards for *.com and *.net

Essentially, catch-all records for unknown domains pointed to a search website run by Verisign. The search website was full of advertisements.
Site Finder

- September 2003: Verisign created DNS wildcards for *.com and *.net
  - Essentially, catch-all records for unknown domains
  - Pointed to a search website run by Verisign
  - Search website was full of advertisements
Site Finder

- September 2003: Verisign created DNS wildcards for *.com and *.net
  - Essentially, catch-all records for unknown domains
  - Pointed to a search website run by Verisign
  - Search website was full of advertisements

- Extremely controversial move
  - Is this DNS hijacking?
  - Definitely abuse of trust by Verisign
  - Site Finder was quickly shut down, lawsuits ensued
Much More to DNS

- Caching: when, where, how much, etc.
Much More to DNS

- Caching: when, where, how much, etc.
- Other uses for DNS (i.e. DNS hacks)
  - Content Delivery Networks (CDNs)
  - Different types of DNS load balancing
  - Dynamic DNS (e.g. for mobile hosts)
Much More to DNS

- Caching: when, where, how much, etc.
- Other uses for DNS (i.e. DNS hacks)
  - Content Delivery Networks (CDNs)
  - Different types of DNS load balancing
  - Dynamic DNS (e.g. for mobile hosts)
- DNS and botnets
Much More to DNS

- Caching: when, where, how much, etc.
- Other uses for DNS (i.e. DNS hacks)
  - Content Delivery Networks (CDNs)
  - Different types of DNS load balancing
  - Dynamic DNS (e.g. for mobile hosts)
- DNS and botnets
- Politics and growth of the DNS system
  - Governance
  - New TLDs (.xxx, .biz), eliminating TLDs altogether
  - Copyright, arbitration, squatting, typo-squatting
Outline

- DNS
- NAT
- Other middleboxes
The IPv4 Shortage

- Problem: consumer ISPs typically only give one IP address per-household
  - Additional IPs cost extra
  - More IPs may not be available
The IPv4 Shortage

- Problem: consumer ISPs typically only give one IP address per-household
  - Additional IPs cost extra
  - More IPs may not be available
- Today’s households have more networked devices than ever
  - Laptops and desktops
  - TV, bluray players, game consoles
  - Tablets, smartphones, eReaders
The IPv4 Shortage

- Problem: consumer ISPs typically only give one IP address per-household
  - Additional IPs cost extra
  - More IPs may not be available
- Today’s households have more networked devices than ever
  - Laptops and desktops
  - TV, bluray players, game consoles
  - Tablets, smartphones, eReaders
- How to get all these devices online?
Private IP Networks

- Idea: create a range of private IPs that are separate from the rest of the network
  - Use the private IPs for internal routing
  - Use a special router to bridge the LAN and the WAN

- Properties of private IPs
  - Not globally unique
  - Usually taken from non-routable IP ranges (why?)

- Typical private IP ranges
  - 10.0.0.0 – 10.255.255.255
  - 172.16.0.0 – 172.31.255.255
  - 192.168.0.0 – 192.168.255.255
Private Networks

192.168.0.0

Private Network

192.168.0.1
192.168.0.2

Internet

66.31.210.69
Private Networks

192.168.0.1
192.168.0.2

Private Network

192.168.0.1
192.168.0.2

Private Network

192.168.0.0
66.31.210.69

Internet

192.168.0.0
Private Networks

192.168.0.0

Private Network

192.168.0.1
192.168.0.2

Private Network

192.168.0.1
192.168.0.2

Internet

66.31.210.69
192.168.0.0

NAT
Network Address Translation (NAT)

- NAT allows hosts on a private network to communicate with the Internet
  - Warning: connectivity is not seamless

- Special router at the boundary of a private network
  - Replaces internal IPs with external IP
    - This is “Network Address Translation”
  - May also replace TCP/UDP port numbers

- Maintains a table of active flows
  - Outgoing packets initialize a table entry
  - Incoming packets are rewritten based on the table
Basic NAT Operation

Private Network: 192.168.0.1
Internet: 66.31.210.69
Public IP Address: 74.125.228.67
Basic NAT Operation

Private Network

Source: 192.168.0.1
Dest: 74.125.228.67

Internet

192.168.0.1
66.31.210.69
74.125.228.67
Basic NAT Operation

Private Network

Source: 192.168.0.1
Dest: 74.125.228.67

Internet

<table>
<thead>
<tr>
<th>Private Address</th>
<th>Public Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.1:2345</td>
<td>74.125.228.67:80</td>
</tr>
</tbody>
</table>

192.168.0.1

66.31.210.69

74.125.228.67
Basic NAT Operation

Private Network

Source: 192.168.0.1
Dest: 74.125.228.67

Private Address
192.168.0.1:2345

Public Address
74.125.228.67:80

Internet

Source: 66.31.210.69
Dest: 74.125.228.67

Private Address
66.31.210.69

Public Address
74.125.228.67
Basic NAT Operation

Private Network

Source: 192.168.0.1
Dest: 74.125.228.67

Internet

Source: 66.31.210.69
Dest: 74.125.228.67

Private Address
Public Address

192.168.0.1:2345 74.125.228.67:80

192.168.0.1 66.31.210.69 74.125.228.67

Source: 74.125.228.67
Dest: 66.31.210.69
Basic NAT Operation

Private Network

Source: 192.168.0.1
Dest: 74.125.228.67

Private Address

192.168.0.1:2345

Public Address

74.125.228.67:80

Internet

Source: 66.31.210.69
Dest: 74.125.228.67

Source: 74.125.228.67
Dest: 192.168.0.1

192.168.0.1

66.31.210.69

Source: 74.125.228.67
Dest: 66.31.210.69

74.125.228.67
Advantages of NATs

- Allow multiple hosts to share a single public IP
Advantages of NATs

- Allow multiple hosts to share a single public IP
- Allow migration between ISPs
  - Even if the public IP address changes, you don’t need to reconfigure the machines on the LAN
Advantages of NATs

- Allow multiple hosts to share a single public IP
- Allow migration between ISPs
  - Even if the public IP address changes, you don’t need to reconfigure the machines on the LAN
- Load balancing
  - Forward traffic from a single public IP to multiple private hosts
## Natural Firewall

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Private Address</td>
<td>Public Address</td>
</tr>
<tr>
<td>192.168.0.1</td>
<td>66.31.210.69</td>
</tr>
<tr>
<td></td>
<td>74.125.228.67</td>
</tr>
</tbody>
</table>
Natural Firewall

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Private Address</strong></td>
<td><strong>Public Address</strong></td>
</tr>
<tr>
<td>192.168.0.1</td>
<td></td>
</tr>
<tr>
<td>66.31.210.69</td>
<td></td>
</tr>
</tbody>
</table>

Source: 74.125.228.67
Dest: 192.168.0.1
# Natural Firewall

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Private Address</td>
<td>Public Address</td>
</tr>
<tr>
<td>192.168.0.1</td>
<td>66.31.210.69</td>
</tr>
<tr>
<td></td>
<td>74.125.228.67</td>
</tr>
</tbody>
</table>
## Natural Firewall

### Table: Private Network vs. Internet

<table>
<thead>
<tr>
<th>Private Address</th>
<th>Public Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.1</td>
<td>66.31.210.69</td>
</tr>
<tr>
<td>74.125.228.67</td>
<td>Source: 74.125.228.67, Dest: 66.31.210.69</td>
</tr>
</tbody>
</table>
Natural Firewall

Private Network | Internet
---|---
Private Address | Public Address

| Source: 74.125.228.67 | Dest: 66.31.210.69

192.168.0.1 | 66.31.210.69 | 74.125.228.67
# Natural Firewall

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Private Address</strong></td>
<td><strong>Public Address</strong></td>
</tr>
<tr>
<td>192.168.0.1</td>
<td>66.31.210.69</td>
</tr>
<tr>
<td>66.31.210.69</td>
<td>74.125.228.67</td>
</tr>
</tbody>
</table>
Concerns About NAT

- Performance/scalability issues
  - Per flow state!
  - Modifying IP and Port numbers means NAT must recompute IP and TCP checksums
Concerns About NAT

- Performance/scalability issues
  - Per flow state!
  - Modifying IP and Port numbers means NAT must recompute IP and TCP checksums
- Breaks the layered network abstraction
Concerns About NAT

- Performance/scalability issues
  - Per flow state!
  - Modifying IP and Port numbers means NAT must recompute IP and TCP check sums

- Breaks the layered network abstraction

- Breaks end-to-end Internet connectivity
  - 192.168.*.* addresses are private
  - Cannot be routed to on the Internet
  - Problem is worse when both hosts are behind NATs
Concerns About NAT

- Performance/scalability issues
  - Per flow state!
  - Modifying IP and Port numbers means NAT must recompute IP and TCP checksums

- Breaks the layered network abstraction

- Breaks end-to-end Internet connectivity
  - 192.168.*.* addresses are private
  - Cannot be routed to on the Internet
  - Problem is worse when **both** hosts are behind NATs

- What about IPs embedded in data payloads?
## Port Forwarding

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Private Address</td>
<td>Public Address</td>
</tr>
<tr>
<td>192.168.0.1:7000</td>
<td><em>.</em>.<em>.</em>:***</td>
</tr>
</tbody>
</table>

- **Private Network**: 192.168.0.1
- **Internet**: 66.31.210.69
- **Public IP**: 74.125.228.67
Port Forwarding

<table>
<thead>
<tr>
<th>Private Address</th>
<th>Public Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.1:7000</td>
<td><em>:</em>:<em>:</em></td>
</tr>
</tbody>
</table>

**Diagram:**
- **Private Network:** 192.168.0.1
- **Internet:** 66.31.210.69
- **Internet** to **Private Network**:
  - **Source:** 74.125.228.67:8679
  - **Dest:** 66.31.210.69:7000
Port Forwarding

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Private Address</strong></td>
<td><strong>Public Address</strong></td>
</tr>
</tbody>
</table>
| 192.168.0.1:7000 | *.*.*.*:*

192.168.0.1  
66.31.210.69  
74.125.228.67

Source: 74.125.228.67:8679  
Dest: 66.31.210.69:7000
Port Forwarding

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Private Address</td>
<td>Public Address</td>
</tr>
<tr>
<td>192.168.0.1:7000</td>
<td><em>.</em>.<em>.</em>:*</td>
</tr>
</tbody>
</table>

192.168.0.1

Source: 74.125.228.67:8679
Dest: 192.168.0.1:7000

66.31.210.69

Source: 74.125.228.67:8679
Dest: 66.31.210.69:7000

74.125.228.67
Problem: How to enable connectivity through NATs?
Problem: How to enable connectivity through NATs?
Hole Punching

Problem: How to enable connectivity through NATs?

NAT 1

192.168.0.1

66.31.210.69

NAT 2

192.168.0.2

59.1.72.13
Hole Punching

- Problem: How to enable connectivity through NATs?

- Two application-level protocols for hole punching
  - STUN
  - TURN
STUN

- **Session Traversal Utilities for NAT**
  - Use a third-party to echo your global IP address
  - Also used to probe for symmetric NATs/firewalls
    - i.e. are external ports open or closed?

192.168.0.1

66.31.210.69

STUN Server
STUN

- **Session Traversal Utilities for NAT**
  - Use a third-party to echo your global IP address
  - Also used to probe for symmetric NATs/firewalls
    - i.e. are external ports open or closed?

---

**What is my global IP address?**

- 192.168.0.1
- 66.31.210.69

STUN Server
**STUN**

- **Session Traversal Utilities for NAT**
  - Use a third-party to echo your global IP address
  - Also used to probe for symmetric NATs/firewalls
    - i.e. are external ports open or closed?

**What is my global IP address?**

```
192.168.0.1
```

```
66.31.210.69
```

**STUN Server**
**STUN**

- **Session Traversal Utilities for NAT**
  - Use a third-party to echo your global IP address
  - Also used to probe for symmetric NATs/firewalls
    - i.e. are external ports open or closed?

---

**What is my global IP address?**

- **Please echo my IP address**
  - 192.168.0.1
  - 66.31.210.69

**STUN Server**
**STUN**

- **Session Traversal Utilities for NAT**
  - Use a third-party to echo your global IP address
  - Also used to probe for symmetric NATs/firewalls
    - i.e. are external ports open or closed?

What is my global IP address?

192.168.0.1

Please echo my IP address

66.31.210.69

STUN Server
Session Traversal Utilities for NAT

- Use a third-party to echo your global IP address
- Also used to probe for symmetric NATs/firewalls
  - i.e. are external ports open or closed?

What is my global IP address?

192.168.0.1

Your IP is 66.31.210.69
Session Traversal Utilities for NAT
- Use a third-party to echo your global IP address
- Also used to probe for symmetric NATs/firewalls
  - i.e. are external ports open or closed?

What is my global IP address?

Your IP is 66.31.210.69
Problems With STUN

- Only useful in certain situations
  - One peer is behind a symmetric NAT
  - Both peers are behind partial NATs
- Not useful when both peers are fully behind full NATs

- NAT 1: 192.168.0.1
  - 66.31.210.69

- NAT 2: 192.168.0.2
  - 59.1.72.13
Traversal Using Relays around NAT
TURN

- Traversal Using Relays around NAT

NAT 1
- 192.168.0.1
- 66.31.210.69

NAT 2
- 192.168.0.2
- 59.1.72.13

TURN Server
Traversing Using Relays around NAT

NAT 1

192.168.0.1

66.31.210.69

NAT 2

192.168.0.2

59.1.72.13

TURN Server
**TURN**

- **Traversal Using Relays around NAT**

```
<table>
<thead>
<tr>
<th>NAT 1</th>
<th>NAT 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.1</td>
<td>192.168.0.2</td>
</tr>
<tr>
<td>66.31.210.69</td>
<td>59.1.72.13</td>
</tr>
<tr>
<td>192.168.0.1:7000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TURN Server</td>
</tr>
</tbody>
</table>
```

TURN

- Traversal Using Relays around NAT

NAT 1
- 192.168.0.1
- 66.31.210.69
- 192.168.0.1:7000

NAT 2
- 192.168.0.2
- 59.1.72.13

TURN Server

192.168.0.1

192.168.0.2
Traversal Using Relays around NAT

NAT 1

Please connect to me on 66.31.210.69:7000

NAT 2

59.1.72.13

192.168.0.2

TURN Server
TURN

- Traversal Using Relays around NAT

**NAT 1**
- 192.168.0.1
- 192.168.0.1:7000
- 66.31.210.69

**NAT 2**
- Please connect to me on 66.31.210.69:7000

TURN Server
**TURN**

- **Traversal Using Relays around NAT**

![Diagram showing TURN traversal using relays around NAT with IP addresses and TURN server](image-url)
**TURN**

- **Traversal Using Relays around NAT**

**Diagram:**
- **NAT 1**
  - 192.168.0.1
  - 192.168.0.1:7000
  - 66.31.210.69

- **NAT 2**
  - 192.168.0.2
  - 192.168.0.2:7000
  - 59.1.72.13

**TURN Server**
Outline

- DNS
- NAT
- Other middleboxes
Firewall

- A device that blocks traffic according to a set of rules
  - Why?
  - Services with vulnerabilities turned on by default
  - ISP policy forbidding certain traffic due to ToS

- Typically specified using a 5-tuple
  - E.g., block outbound SMTP; block inbound SQL server reqs

- GFC (Great Firewall of China)
  - Known to block based on IP, filter DNS requests, etc
Web caching

- ISP installs cache near network edge that caches copies of Web pages
  - Why?
  - **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
  - **Cost**: “free” for the ISP to serve from inside the network

- Limitations
  - Much of today’s content is not static (why does this matter?)
  - Content ownership
  - Potential privacy issues
  - Long tail of content popularity
Web caching

- ISP installs cache near network edge that caches copies of Web pages
  - Why?
  - **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
  - **Cost**: “free” for the ISP to serve from inside the network
Web caching

- ISP installs cache near network edge that caches copies of Web pages

  - Why?
  - **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
  - **Cost**: “free” for the ISP to serve from inside the network
Web caching

- ISP installs cache near network edge that caches copies of Web pages
- Why?
- **Performance:** Content is closer to clients, TCP will perform better with lower RTTs
- **Cost:** “free” for the ISP to serve from inside the network
Web caching

- ISP installs cache near network edge that caches copies of Web pages
  - Why?
  - **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
  - **Cost**: “free” for the ISP to serve from inside the network
Web caching

- ISP installs cache near network edge that caches copies of Web pages
- Why?
- **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
- **Cost**: “free” for the ISP to serve from inside the network
ISP installs cache near network edge that caches copies of Web pages

- Why?
- **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
- **Cost**: “free” for the ISP to serve from inside the network
Web caching

- ISP installs cache near network edge that caches copies of Web pages
  - Why?
  - **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
  - **Cost**: “free” for the ISP to serve from inside the network
Web caching

- ISP installs cache near network edge that caches copies of Web pages
  - Why?
  - **Performance**: Content is closer to clients, TCP will perform better with lower RTTs
  - **Cost**: “free” for the ISP to serve from inside the network
Proxying

- **Non-split connections**
  - Like NAT, but IP address is no longer the one assigned to you

- **Split connections**
  - Middlebox maintains two flows: C-M and M-S
  - Can be done transparently
    - How?
Proxying

- Non-split connections
  - Like NAT, but IP address is no longer the one assigned to you

- Split connections
  - Middlebox maintains two flows: C-M and M-S
  - Can be done transparently
    - How?
Proxying

- Advantages
  - RTT is lower on each end
  - Can use different MTUs
  - Particularly useful in cell ntwks

- Disadvantages
  - Extra delay can be bad for small flows
  - Buffering/state makes it potentially costly
Shaping

- ISPs are often charged according to 95% model
  - Internet usage is very “peaky”, e.g., at 5pm, or when House of Cards season 2 is released

- To control costs, ISPs such as Rogers shape client traffic
  - Time-of day
  - Traffic type

- Common implementations
  - Token Bucket (see next deck)
  - RSTs

![Graph showing 95% throughput samples](image)
Shaping

- ISPs are often charged according to 95% model
  - Internet usage is very “peaky”, e.g., at 5pm, or when House of Cards season 2 is released

- To control costs, ISPs such as Rogers *shape* client traffic
  - Time-of day
  - Traffic type

- Common implementations
  - Token Bucket (see next deck)
  - RSTs
Shaping

- ISPs are often charged according to 95% model
  - Internet usage is very “peaky”, e.g., at 5pm, or when House of Cards season 2 is released

- To control costs, ISPs such as Rogers \textit{shape} client traffic
  - Time-of-day
  - Traffic type

- Common implementations
  - Token Bucket (see next deck)
  - RSTs
Shaping

- ISPs are often charged according to 95% model
  - Internet usage is very “peaky”, e.g., at 5pm, or when House of Cards season 2 is released

- To control costs, ISPs such as Rogers shape client traffic
  - Time-of day
  - Traffic type

- Common implementations
  - Token Bucket (see next deck)
  - RSTs
Shaping

- ISPs are often charged according to 95% model
  - Internet usage is very “peaky”, e.g., at 5pm, or when House of Cards season 2 is released

- To control costs, ISPs such as Rogers shape client traffic
  - Time-of day
  - Traffic type

- Common implementations
  - Token Bucket (see next deck)
  - RSTs