CSE 127 Computer Security

Stefan Savage, Fall 2025, Lecture 12

Network Security II: DNS, perimeter defense, and denial-of-service

Today

The Domain Name System

- Another place where names at different layers can bound
- The target of quite a few attacks

Network perimeter defenses

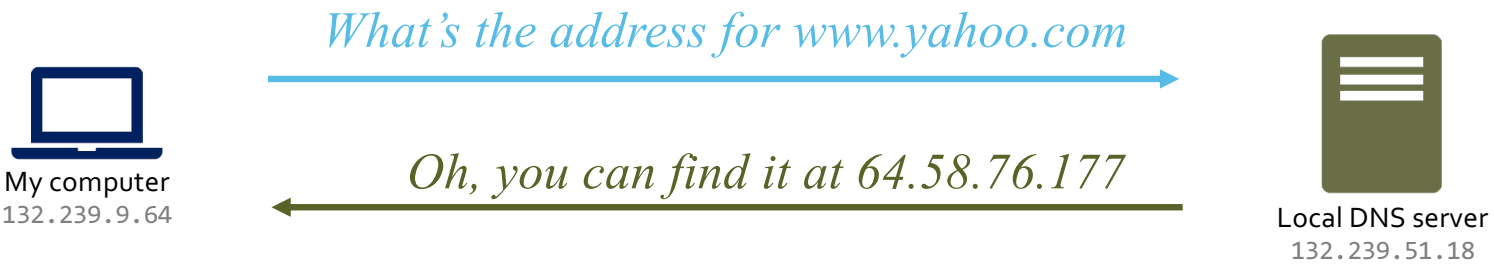
- Firewalls, IDS/IPS

Denial of service

- Network attacks on availability

Remember this? How did this work?

Where is www.yahoo.com?



Ignore for now:
- How did you know the address of the Local DNS server?
- How did you send a message to it?
- How did the Local DNS server know the answer?

Domain Name System

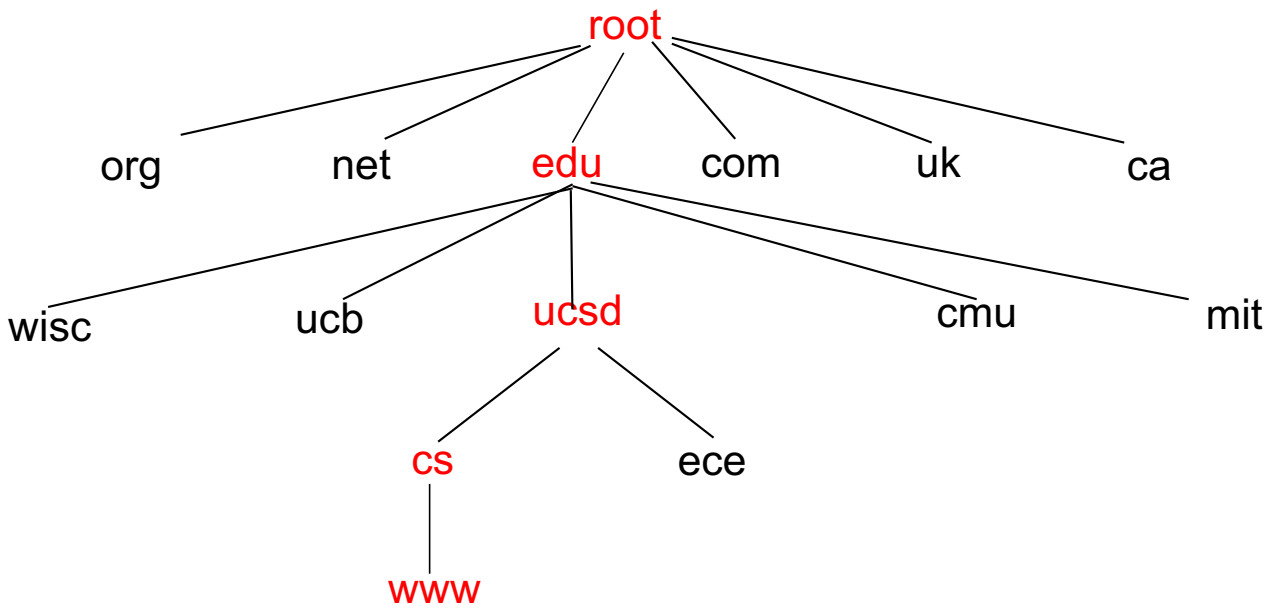
- We humans do not tend to remember 32bit IP numbers…

- Solution: domain names
  - Human readable identifiers (e.g., www.cs.ucsd.edu, google.com, etc)

- Problem: how to map DNS names to IP addresses?
  - In the old days we had a big file – *literally* (download from sri-nic.arpa)
  - Today we use a distributed name servers called the Domain Name System (DNS)

Domain Name System (DNS)

- Hierarchical Name Space



DNS Root Name Servers

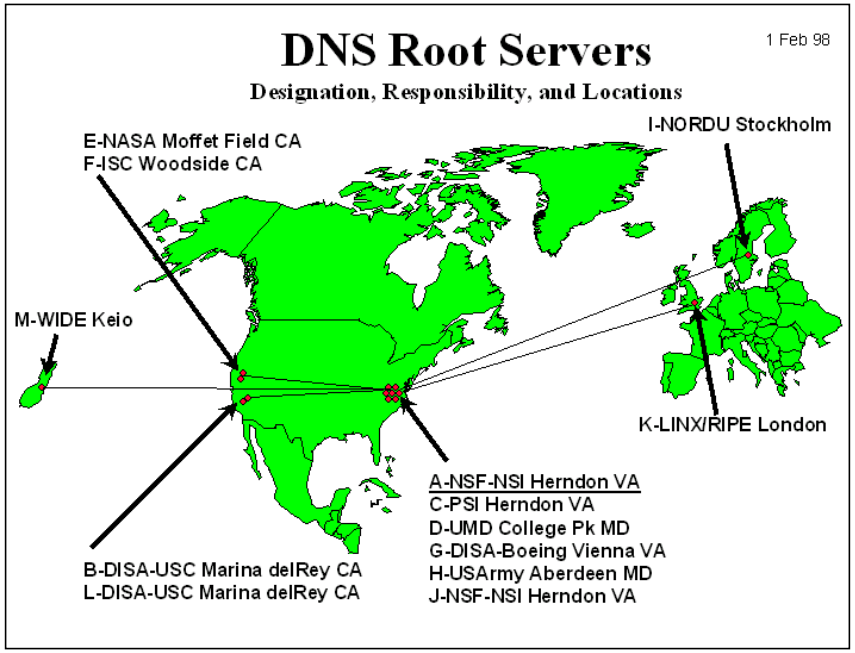
Hierarchical service

- 13 root name servers for top-level domains (13 AP addresses)

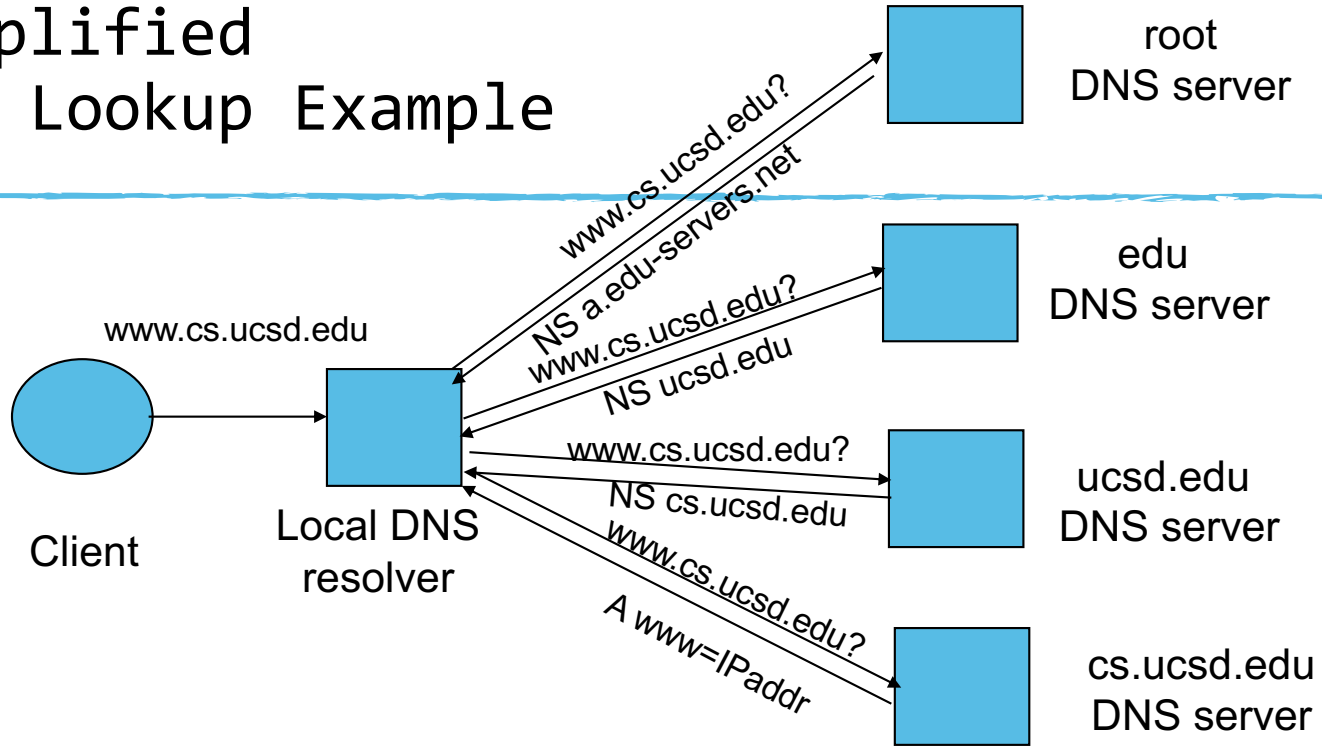
  Hardcoded into all systems

  Choose one at random

  - Note many hundreds of replicas, not just 13 individual servers
- Authoritative name servers for subdomains
- Local name resolvers (also called recursive resolvers) contact authoritative servers when they do not know a name



Simplified DNS Lookup Example



DNS record types (partial list):

- NS: name server (points to other server's DNS name)
- A: address record (contains IP address)
- MX: address in charge of handling email
- TXT: generic text (e.g. used to distribute site public keys (DKIM))

Caching

- DNS **responses are cached**
  - Quick response for repeated translations
  - Useful for finding nameservers as well as IP addresses
    NS records for domains ("hey, for any info about ucsd.edu ask ns1.ucsd.edu")

- DNS **negative queries** are also cached
  - Save time for nonexistent sites, e.g. misspelling ("hey, aamazon.com doesn't exist")

- Cached data periodically times out
  - Lifetime in seconds (TTL) of data controlled by owner of data
  - TTL passed with every record, delete cached entry after TTL expires

DNS cache poisoning

Basic idea:

- If I can convince a DNS resolver to **cache** a bad mapping (e.g., www.cs.ucsd.edu points to 127.0.0.1) then everyone who uses it for www.cs.ucsd.edu will get that incorrect resolution
- Can then be used for fraud, man-in-the-middle attacks, etc

Used in lots of attacks

- January 2005, the domain name for a large New York ISP, Panix, was hijacked to a site in Australia.
- In November 2004, Google and Amazon users were sent to Med Network Inc., an online pharmacy
- In March 2003, a group dubbed the "Freedom Cyber Force Militia" hijacked visitors to the Al-Jazeera Web site and presented them with the message "God Bless Our Troops"
- 2000 campaign: Hilary2000.org -> hilaryno.com
- Gazillions of others… super common

Governments actors too

Both attackers and victims

Type	Hij.	Targeted Domain Information CC	Domain	Sub.	Cross Ref pDNS	crt	Attacker Infra. (Transient) IP	ASN	CC	Legitimate Infra. (Stable) ASNs	CCs
T1	May'18	AE	mofa.gov.ae	webmail	✔	✔	146.185.143.158	14061	NL	[5384,202024]	[AE]
T1	Sep'18	AE	adpolice.gov.ae	advpn	✔	✔	185.20.187.8	50673	NL	[5384]	[AE]
T1*	Sep'18	AE	apc.gov.ae	mail	✖	✔	185.20.187.8	50673	NL	[5384]	[AE]
T2	Sep'18	AE	mgov.ae	mail	✔	✔	185.20.187.8	50673	NL	[202024]	[AE]
T1	Jan'18	AL	e-albania.al	owa	✔	✔	185.15.247.140	24961	DE	[5576]	[AL]
T2	Nov'18	AL	asp.gov.al	mail	✔	✔	199.247.3.191	20473	DE	[201524]	[AL]
T1	Nov'18	AL	shish.gov.al	mail	✔	✔	37.139.11.155	14061	NL	[5576]	[AL]
T1	Dec'18	CY	govcloud.gov.cy	personal	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
P-IP	Dec'18	CY	owa.gov.cy	.	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
T1	Dec'18	CY	webmail.gov.cy	.	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
P-IP	Jan'19	CY	cyta.com.cy	mbox	✔	✔	178.62.218.244	14061	NL	–	–
T1	Jan'19	CY	sslvpn.gov.cy	.	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
T1	Feb'19	CY	defa.com.cy	mail	✔	✔	108.61.123.149	20473	FR	[35432]	[CY]
T1	Nov'18	EG	mfa.gov.eg	mail	✔	✔	188.166.119.57	14061	NL	[37066]	[EG]
T2	Nov'18	EG	mod.gov.eg	mail	✔	✔	188.166.119.57	14061	NL	[25576]	[EG]
T2	Nov'18	EG	nmi.gov.eg	mail	✔	✔	188.166.119.57	14061	NL	[31065]	[EG]
T1	Nov'18	EG	petroleum.gov.eg	mail	✔	✔	206.221.184.133	20473	US	[24835,37191]	[EG]
T1	Apr'19	GR	kyvernisi.gr	mail	✔	✔	95.179.131.225	20473	NL	[35506]	[GR]
T1	Apr'19	GR	mfa.gr	pop3	✔	✔	95.179.131.225	20473	NL	[35506,6799]	[GR]
T2	Sep'18	IQ	mofa.gov.iq	mail	✔	✔	82.196.9.10	14061	NL	[50710]	[IQ]
P-IP	Nov'18	IQ	inc-vrdl.iq	.	✔	✔	199.247.3.191	20473	DE	[50710]	[IQ]
P-NS	Dec'18	JO	gid.gov.jo	.	✔	✔	139.162.144.139	63949	DE	–	–
P-NS	Dec'20	KG	fiu.gov.kg	mail	✔	✔	178.20.41.140	48282	RU	–	–
T1	Dec'20	KG	invest.gov.kg	mail	✔	✔	94.103.90.182	48282	RU	[39659]	[KG]
T1	Dec'20	KG	mfa.gov.kg	mail	✔	✔	94.103.91.159	48282	RU	[39659]	[KG]
P-NS	Jan'21	KG	infocom.kg	mail	✔	✔	195.2.84.10	48282	RU	–	–
T1	Dec'17	KW	csb.gov.kw	mail	✔	✔	82.102.14.232	20860	GB	[6412]	[KW]
P-IP	Dec'18	KW	dgca.gov.kw	mail	✔	✔	185.15.247.140	24961	DE	–	–
T1*	Apr'19	KW	moh.gov.kw	webmail	✖	✔	91.132.139.200	9009	AT	[21050]	[KW]
T2	May'19	KW	kotc.com.kw	mail2010	✔	✔	91.132.139.200	9009	US	[57719]	[KW]
P-IP	Nov'18	LB	finance.gov.lb	webmail	✔	✔	185.20.187.8	50673	NL	–	–
P-IP	Nov'18	LB	mea.com.lb	memail	✔	✔	185.20.187.8	50673	NL	–	–
T1	Nov'18	LB	medgulf.com.lb	mail	✔	✔	185.161.209.147	50673	NL	[31126]	[LB]
T1	Nov'18	LB	pcm.gov.lb	mail1	✔	✔	185.20.187.8	50673	NL	[51167]	[DE]
P-IP	Oct'18	LY	embassy.ly	.	✔	✖	188.166.119.57	14061	NL	–	–
P-NS	Oct'18	LY	foreign.ly	.	✔	✔	188.166.119.57	14061	NL	–	–
T1	Oct'18	LY	noc.ly	mail	✔	✔	188.166.119.57	14061	NL	[37284]	[LY]
T1	Jan'18	NL	ocom.com	connect	✔	✔	147.75.205.145	54825	US	[60781]	[NL]
P-NS	Jan'19	SE	netnod.se	dnsnodeapi	✔	✔	139.59.134.216	14061	DE	–	–
T1	Mar'19	SY	syriatel.sy	mail	✔	✔	45.77.137.65	20473	NL	[29256]	[SY]
P-NS	Dec'18	US	pch.net	keriomail	✔	✔	159.89.101.204	14061	DE	–	–

Table 2: List of 41 domains identified as hijacked between January 2017 and March 2021. Type indicates how we identified the domain. For every domain, we report the time of first hijack, the targeted subdomain, the country associated with the

But how to do it?

Person-in-the-middle network attacker: **easy**

- Observe DNS requests from server
- Send false response to server and block true response

Passive monitor attacker: **easy also**

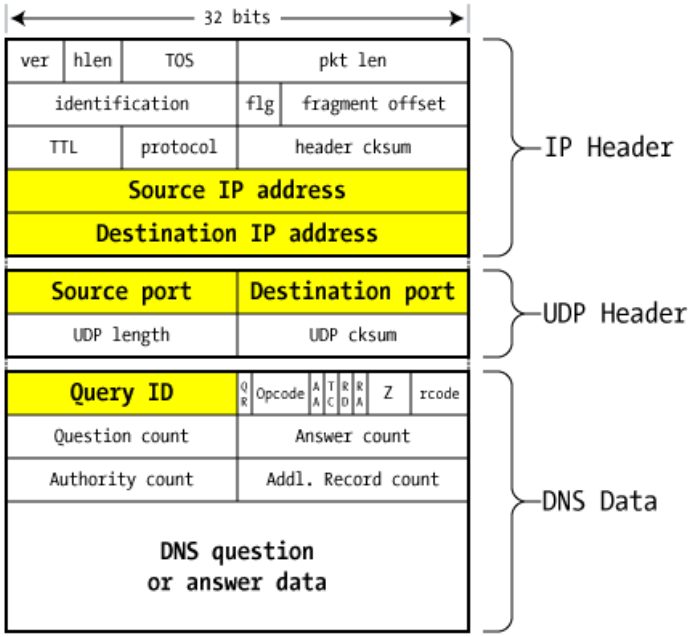
- Observe DNS requests from server
- Send false response to server **before** true response

What about off-path attacker?

- If you know when user is likely to make request, you can flood responses to their server blindly
- But there's a problem – *matching request and response*

DNS Packet

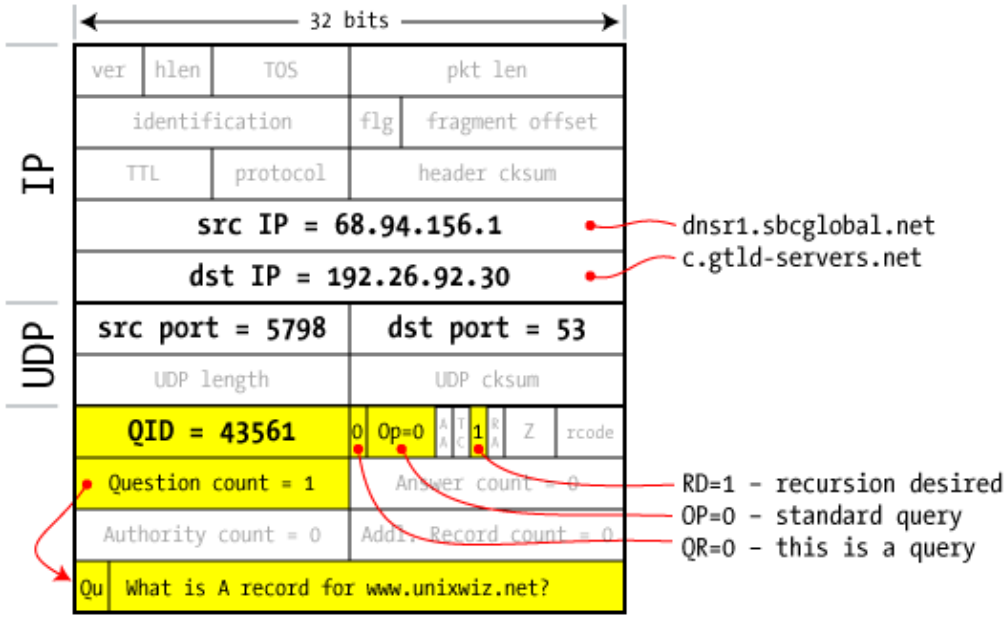
- Query ID:
  - 16 bit random value
  - *Links response to query*



(from Steve Friedl)

Resolver to NS request

- Context
  - SBC Global customer looks up unixwiz.net
  - Goes to their DNS server dnsr1.sbcglobal.net
  - This is the request sent from **that** server to c.gtld-servers.net which can give authoritative answers for domains ending in .net



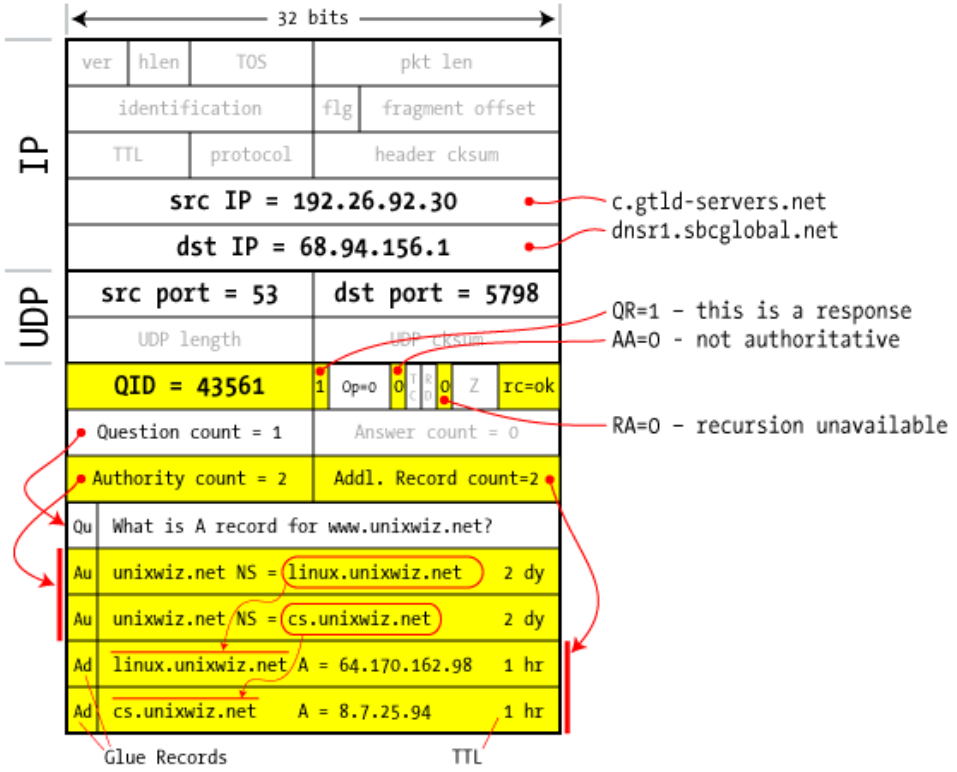
Response to resolver

Response echoes QueryID (requestor will ignore response if it doesn't match)

Response contains name of authoritative nameservers for unixwiz domain

AND

IP addr of those servers (called "glue")



DNS response *additional section*

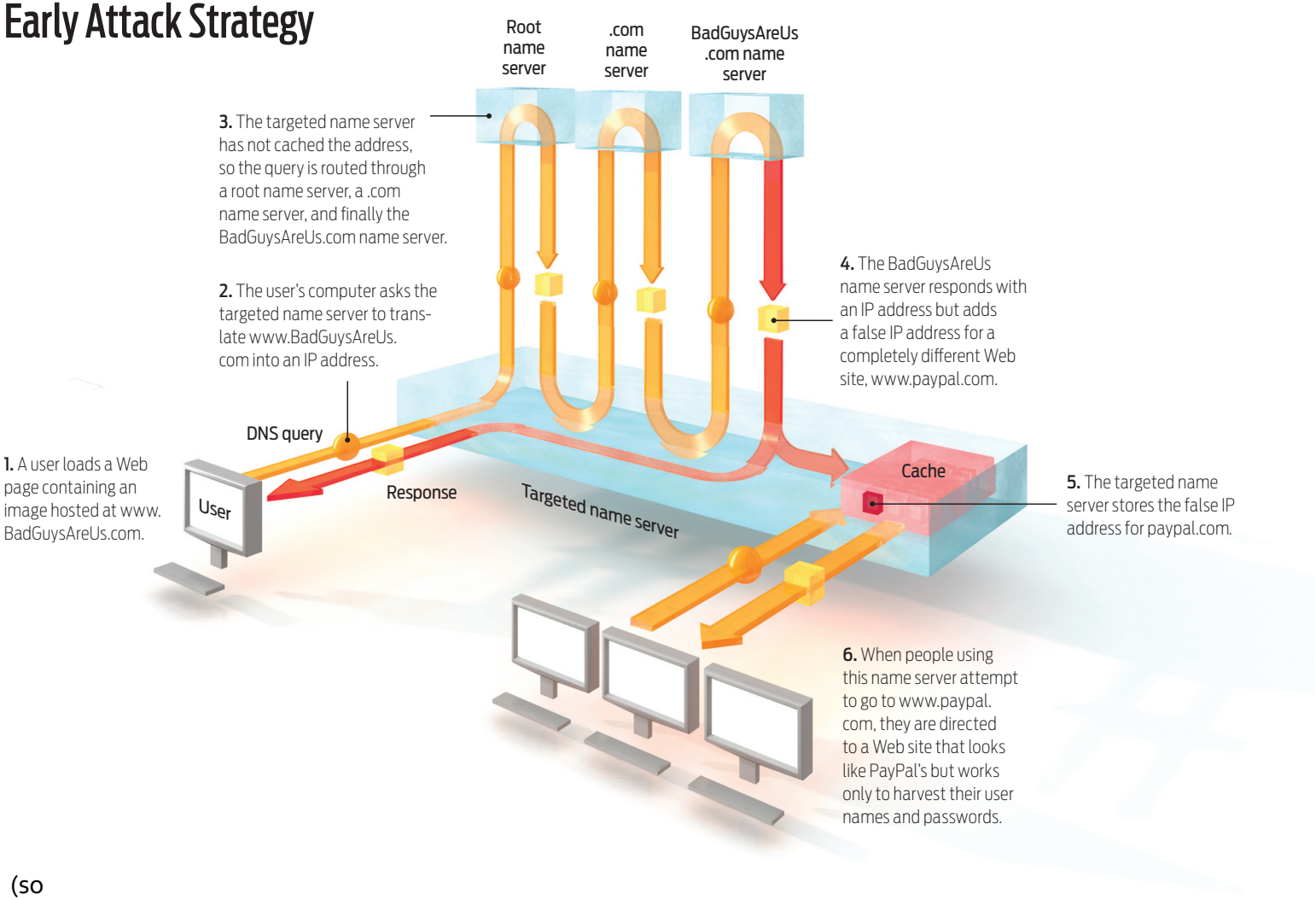
Answers to questions you didn't ask…

- Notably "glue" records
- There is a good reason for it… if I tell you that the nameserver for foo.com is ns1.foo.com… how do you find its IP address?

But this is a problem… what if I run a DNS server for foo.com and when I get asked for the IP address for bar.foo.com, I put some additional stuff in the "additional section" like:

- You can find the IP address for paypal.com at 41.2.6.2 (some address I control)
- And you can also find the IP address for amazon.com and chase.com there too…

Early Attack Strategy

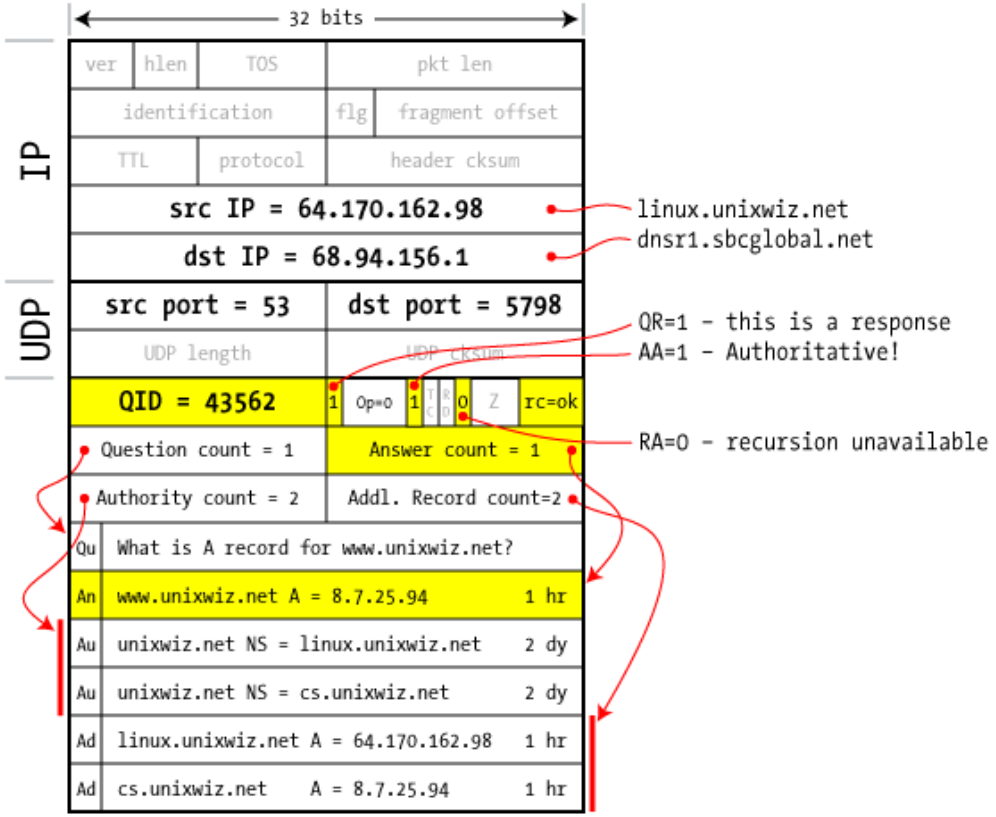


(so

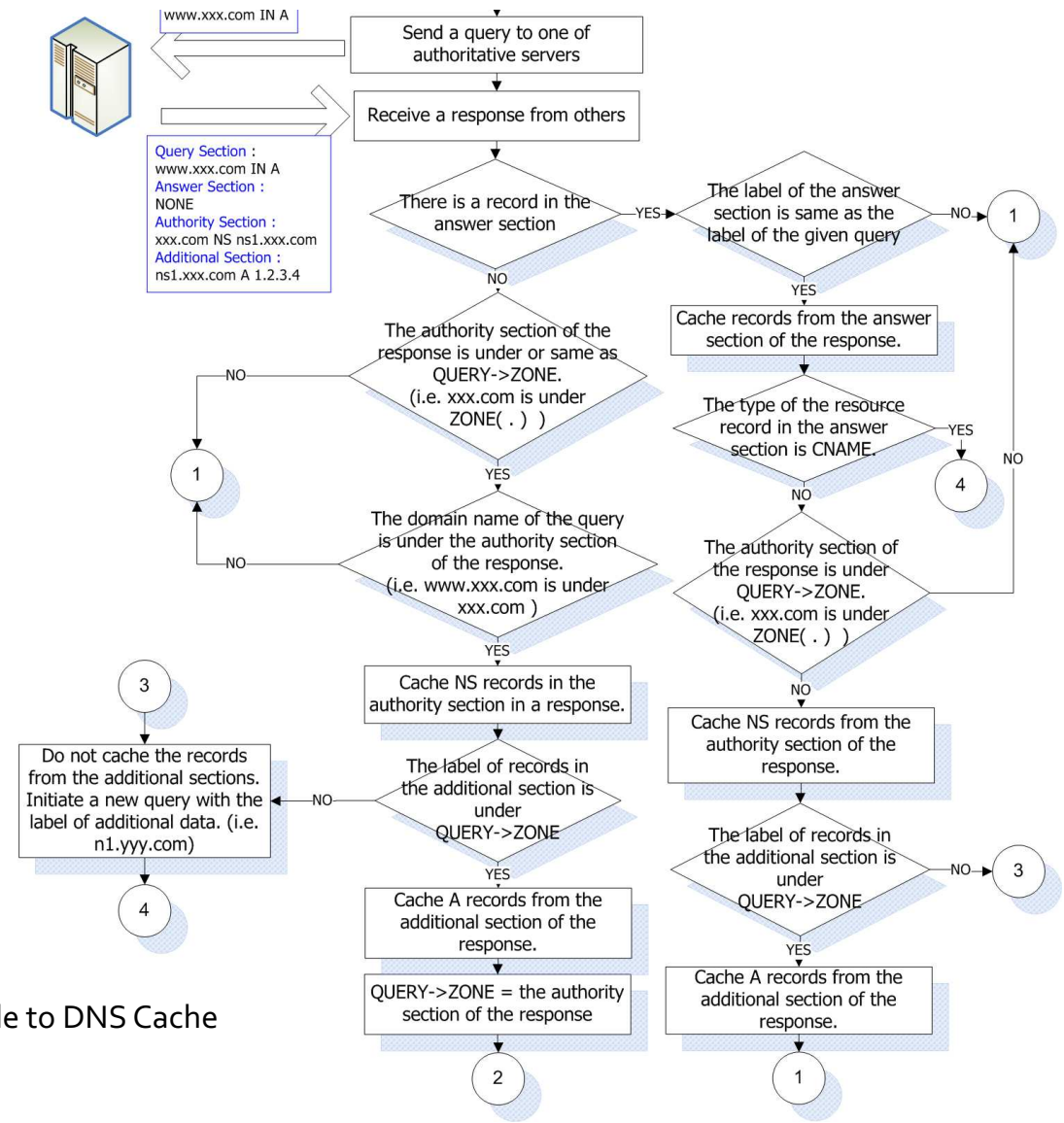
Solution: Bailiwick checking

bailiwick checking:
response is cached **only** if it is within the same domain as the query (i.e. a.com cannot set NS for b.com)

final answer →



Bailiwick Checking Rule from BIND



source: Son and Shmatikov, "The Hitchhiker's Guide to DNS Cache Poisoning" SECURECOMM 2010

But we forgot something…

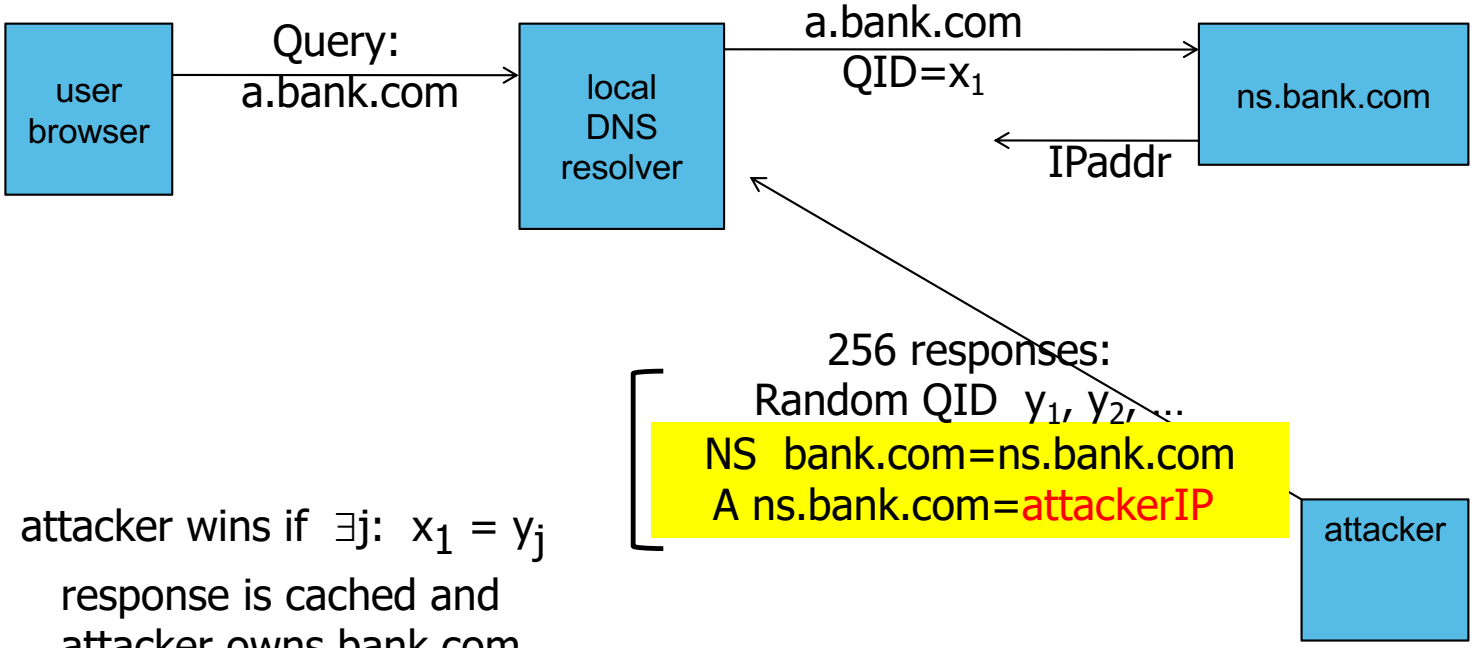
A decade goes by and Dan Kaminsky realizes that the bailiwick checking rule only ***looks like*** it protects us

Unnoticed hole that allows **arbitrary DNS poisoning** at a distance

To fix bug, ***unprecedented*** coordinated global operation to do **secret** mass migration of all major DNS infrastructure (2008)

DNS cache poisoning (a la Kaminsky'08)

- Victim machine visits attacker's web site, downloads Javascript

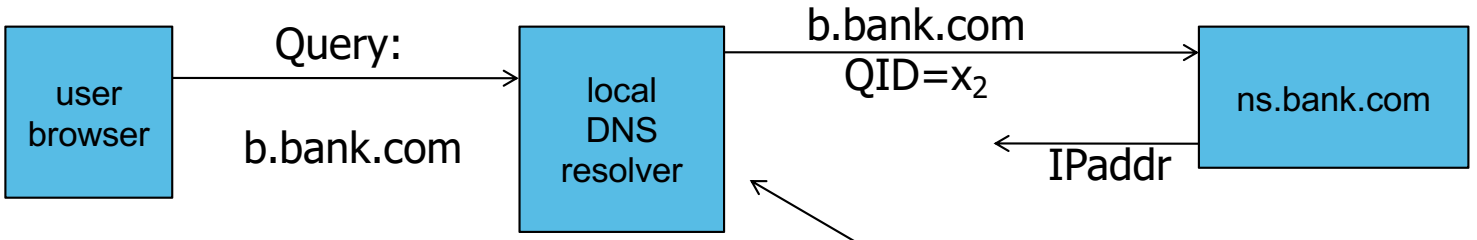


attacker wins if $\exists j$: $x_1 = y_j$

response is cached and attacker owns bank.com

If at first you don't succeed …

- Victim machine visits attacker's web site, downloads Javascript



256 responses:
Random QID y_1, y_2, …
NS bank.com=ns.bank.com
A ns.bank.com=attackerIP

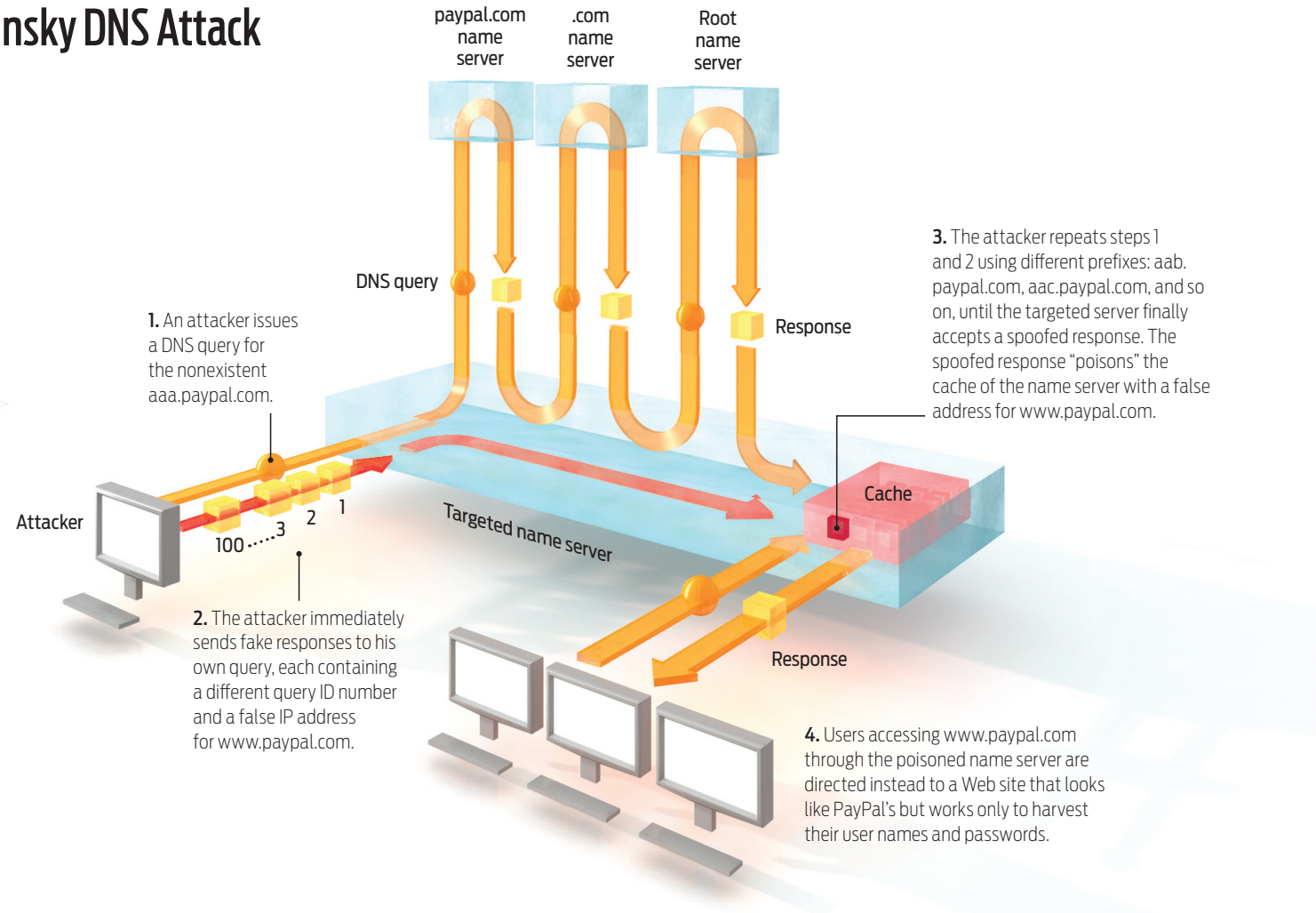
attacker

attacker wins if $\exists j$: $x_2 = y_j$

response is cached and attacker owns bank.com

Only 16bits in QueryID
success after ≈ 256 tries (few minutes)

Kaminsky DNS Attack



BRYAN CHRISTIE DESIGN

Defenses

Increase Query ID size. How? Some approaches in use today:

- Randomize src port and make sure it matches, additional 11 bits
  - Now attack takes hours instead of minutes
- 0x20 encoding – randomly vary capitalization (DNS is case insensitive) check that you get the **same capitalization back**
- DNS cookies
  - New EDNS option to add 64 bit cookie to each query (but server must support)

Try to detect poisoning (e.g., detected failed query matching)

- …and refuse to cache targets
- Ignore responses not directly necessary to query

Rate limit queries for same name

Authenticated requests/responses

- Provided by DNSsec (digital signatures on DNS records) … but few domains use DNSsec

DNS Summary

- Current DNS system does not provide strong evidence binding request to response
- Response can provide more data than was asked for
- Together allows attacker to "poison" DNS and divert traffic to their sites

- This is also why its so important for HTTPS to check cryptographic certificate signatures (i.e., because just because you ask for www.amazon.com doesn't mean you'll get it)

Network Perimeter Defense

Idea: network defenses on "outside" of organization (e.g., between org and Internet)

- Assumptions?

Typical elements

- Firewalls and Application Proxies (e.g., Web Application Firewalls)
- Network Address Translation
- Network Intrusion Detection Systems(NIDS) (and other kinds of network content analysis)

Firewalls

Problem: you'd like to protect or isolate one part of the network from other parts

- Typically: protection your network from the global Internet
- Sometimes
  - Network segmentation: protect one part of your network from another part
  - External hygiene: protect Internet from infected machines in your network

Need to filter or otherwise limit network traffic

Key questions

- What information do you use to filter?
- Where do you do the filtering?

Kinds of Firewalls

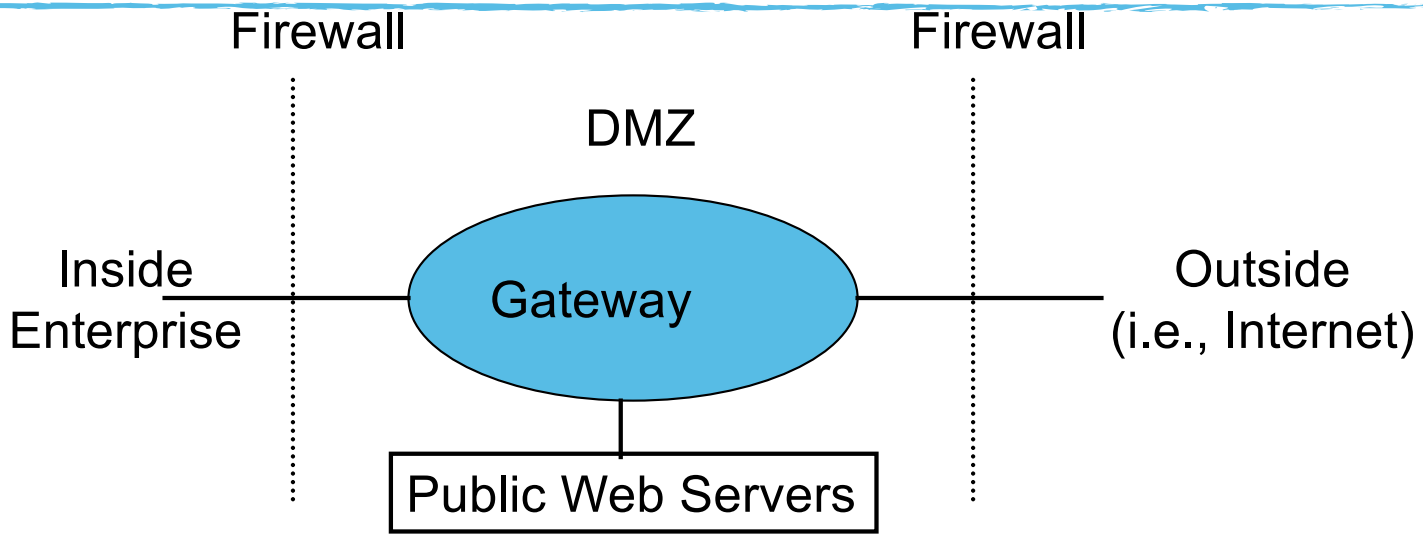
Where they run

- Personal firewalls
  - Run at the end hosts
  - Benefit: has more application/user specific information
- Network firewalls
  - Intercept and evaluate communications from many hosts
  - Deployed "in-line" in network infrastructure (i.e., mediates all communications)

What they filter on

- Packet filters
  - Operate by filtering on packet headers content
- Proxy-based
  - Operates at the level of the application (e.g., HTTP Web proxy)

Network Firewall common deployment strategy



- Filters protect against “bad” communications.
- Protect services offered internally from outside access.
- Provide outside services to hosts located inside.

Access control policies

A firewall tries to enforce and access control policy

- Who is allowed to talk to whom and access what services

Distinguish between inbound and outbound connections

- Inbound: attempts by external users to connect to services on internal machines
- Outbound: attempts by internal users to connect to external services (i.e., out on the Internet)

Conceptually simple access control policy

- Permit users inside to connect to most services (maybe not known bad sites)
- External users are restricted

  Allow connections to services meant to be exposed to the outside (e.g., public web servers, server for receiving mail)

  Deny connections to services not meant to be exposed?

Defaults

How to represent policy?

Default allow

- Permit all services and only specify the ones that you want to exclude

Default deny

- Deny all services and only specify the ones you want to allow

In general, default deny is safer. Why?

- Conservative design
- Flaws in default deny get noticed more quickly
- But… assumptions about how many services there are
  (i.e., this is why many organizations are default allow for **outbound traffic**)

Packet Filtering Firewalls

Define list of access-control rules

Check every packet against rules and forward or drop

Packet filtering firewalls can take advantage of the following information from network and transport layer headers:

- Source IP
- Destination IP
- Source Port
- Destination Port
- Flags (e.g. ACK)

Ports

Ports are used to distinguish applications and services on a machine.

Low numbered ports are often reserved for server listening.

High numbered ports are often assigned for client requests

Imperfect language…

Result: everything gets "tunneled" through ports not blocked by the firewall (e.g., 80, 443)

Port 20 (TCP): FTP data

Port 21 (TCP): FTP control

Port 22 (TCP): ssh

Port 25 (TCP): SMTP (Mail)

Port 80 (TCP): HTTP

Port 123 (UDP): NTP

Port 143 (TCP): IMAP

Port 443 (TCP): HTTPS

Example Firewall policy

Configure: Allow outbound traffic from network 100.64.0.x (private network), deny all else

```
# ufw default deny
# ufw allow from 100.64.0.0/24
# ufw allow ssh
```

Other kinds of typical rules:

- Block incoming HTTPS (port 443) except the IP addresses of company's public Web servers
- Block incoming/outgoing traffic to these known bad address (blocklist)
- Block outgoing packets with source IP address not from company (i.e., forged)

Aside: Stateless rules and TCP

Stateless rules (like the ones of shown)

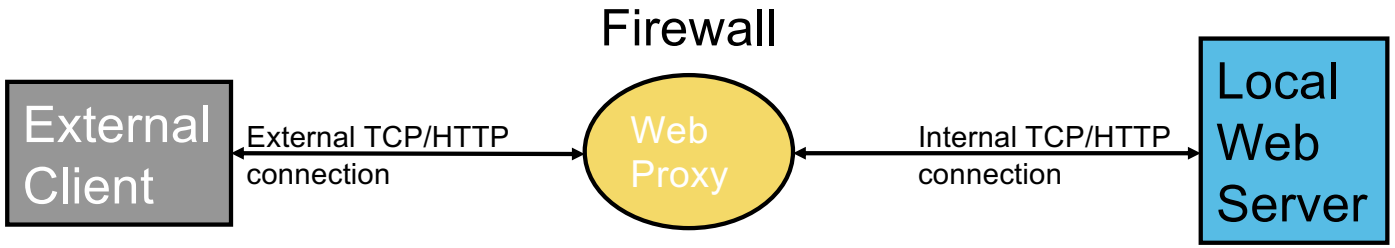
- Treat each packet independently; no knowledge of TCP connection state
- Can't distinguish packets associated with a connection vs those that are not

Solution: some firewalls maintain state about open TCP connections

Allows condition filtering rules of the form:
"if internal machine established outbound TCP connection, then allow inbound reply packets"

But… still only looking at packet headers… limited understanding of app semantics

Proxy-based Firewalls



Proxy acts like *both* a client and a server.

- Note, must terminate connection. Complications for HTTPS/TLS… must take out certificates

More semantics available: able to filter using application-level info

- For example, block based on URL, or on particular javascript strings

Proxies can provide other services too

- Caching, load balancing, etc.

Firewalls Pro/Con

Benefits

- Reduced "attack surface" against external attackers
- Filter out lots of "noise" in network traffic
  (helps focus attention)
- Reduced liability (sense that it is "best common practice")

Costs

- Actual cost: both hardware and administration
- Bottleneck and single point of failure on network
- False sense of security

  Limited language (addresses, ports); doesn't help with worms/viruses, ssh exploits, cross-site scripting, Inside vs outside model is fragile (once an internal host is compromised firewall does no good); What about wireless laptops?

  Modern companies increasingly offer no additional trust to machines inside the firewall
  (so-called "zero trust" architectures)

Network content analysis

Lots of devices want to look at network traffic **content** for security

- Network Intrusion detection/prevention Systems (NIDS/NIPS)
  - Try to find signatures of attacks or malware
- Spam filters
  - Try to detect unwanted e-mail
- Data leakage
  - Try to prevent sensitive information from leaving company
- Traffic differentiation
  - Filter or slow down BitTorrent traffic, Netflix traffic, etc

Doing this as in the network is attractive because its cheaper and easier to manage than putting endpoint monitoring on each host

Challenges

Expensive to look into each packet

- 10Gbps -> ~1M packets per second… ns' per byte
- Also must reassemble pkts into in-order streams
  (e.g., what if signature you are looking for spans two packets)

Network vantage point is imperfect

- What if a session is encrypted?
- Even if its not, what does a packet *mean*?
- Network evasion?

Network evasion

Typically network intrusion detection systems are deployed like firewalls (between internal network and Internet)

Key assumption is that NIDS sees the same traffic as destination host

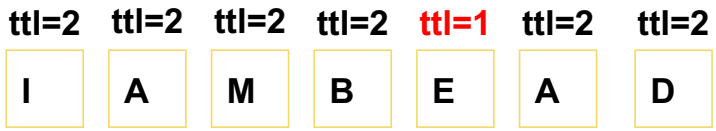
Not quite true…

- Lots of ways to evade a NIDS by exploiting ambiguity

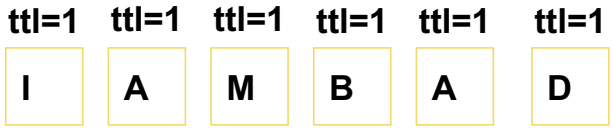
TTL evasion

Suppose destination host is **2 hops** inside network ***after*** NIDS box

This is what NIDS sees (assume each pkt has one byte)

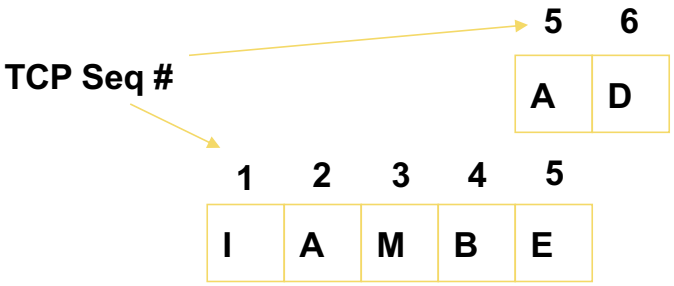


This is what the destination host sees



Sequence # evasion

Suppose attacker sends two packets



What does destination see?

- IAMBED? IAMBAD? IAMBD? Depends on host

Lots of other evasion techniques…

Solution

Protocol normalization

- NIDS **rewrites packets** to remove all ambiguity
  - E.g., all packets have ttl rewritten to reach **any** internal destination
  - E.g., IDS tracks each flow and **does not allow** overlapping packets

Can be very tricky to get right and expensive

- Potentially must buffer large amounts of data
- What if you get seq #2 through #100, but not seq #1?
  - How long do you wait for seg #1 to arrive before you can free that data?
- Tradeoff: when to drop data vs when to buffer

Bottom line

The network vantage point is a very appealing place to implement policy because its central – **everyone has to go through it**!

- But very challenging to infer the semantics of unknown communication
  - Port numbers communication very limited info
  - Parsing content is tough and many ways to evade
  - End-to-end encryption makes it hard to get content (proxies with signing certs)

In spite of this, everyone uses variants of these approaches because they don't have anything better

Denial-of-service

- Attack against *availability*, not confidentiality, integrity, or authenticity
- Two kinds of attacks:
  - **Logic vulnerabilities:** exploit bugs in network code to cause crash

    e.g. Ping-of-Death, Land attacks

    Fix via filtering and patching
  - **Resource consumption**: overwhelm with spurious requests

    e.g. SYN flood, bandwidth overflow

    Much tougher to fix…
    - Same idea, more quantity: Distributed denial-of-service attacks (DDOS)
      - **Lots of hosts** attack a victim at once

Resource consumption of Service

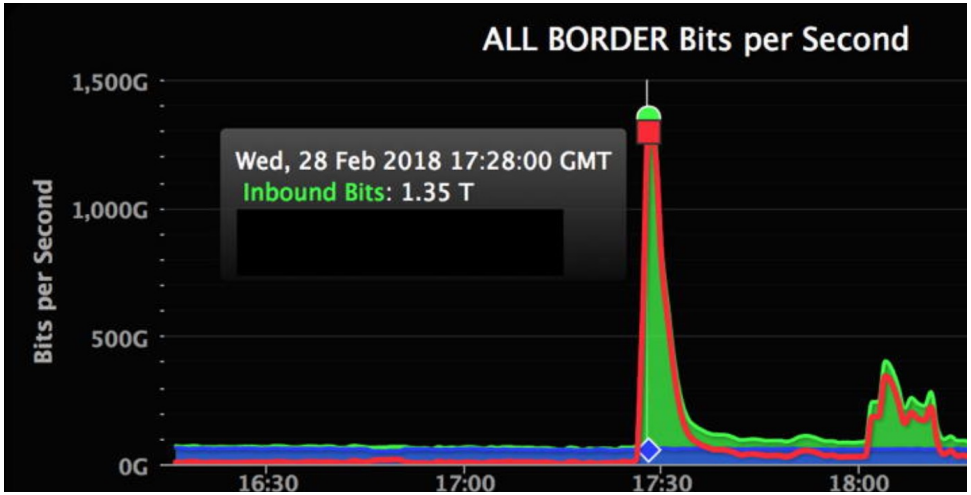
- Server CPU/Memory resources
  - Consumes connection state (e.g. SYN flood)
  - Time to evaluate messages (interrupt livelock)
    Some messages take "slow path" (e.g. invalid ACK)
  - Can cause new connections to be dropped and existing connections to time-out
  - Make DB process lots of queries
    Attack cache – lots of random queries
- Network resources
  - Some routers are packet-per-second limited, FIFO queuing (send small pkts)
  - If attack is greater than forwarding capacity, good data will be dropped (send big pkts)

This is a widespread problem

Most attacks small and focused on individuals or small sites

- Particularly gamers; so-called booter services

But some attacks are huge (e.g., 2018 Attack on GitHub)



What to do?

- Defenses against address spoofing
  - For attacks from randomly spoofed addresses
- Filtering based on attack features or IP address
- Buy more resources

Address spoofing

- **Source Address Validation**: filter packets with clearly bad source addresses
  - **Network egress**: filter outbound packets on a link whose source addresses are not reached using the link as the next hop (i.e., this couldn't be your source address)
    But requires network routers to do "good deeds" for others…
  - **Network ingress**: filter inbound packets whose source address are not in the routing table at all
- **SYN Cookies**
  - Issue: allocating per TCP session state is **expensive** (that's why the SYN flood attack works)
  - Delay allocation of state until after remote host commits to three-way handshake
    Assumption is that spoofing attackers can't complete handshake
  - Send back SYN/ACK packet **without allocating state** on server; server's initial sequence number (ISN) encodes a secret "cookie" that is function of some combination of (src,dst,srcport,dstport and time)
  - Only allocate state when client sends ACK to server's SYN/ACK (using "cookie" to validate)

Packet filtering

- Idea, if there is a common feature to the packet (i.e. "Die, you loser" in the payload, static source port #, odd TCP flags, etc.) then look for those packets and drop them
- If no feature exists then try to find way to add a "good" feature to legitimate flows that complete a 3-way handshake
- Instead of dropping packets, can simply rate-limit packets that are suspicious

- Third-party services will offer these capabilities on your behalf in the network
  - Like dialysis for Internet traffic – route network traffic to devices designed to filter out bad packets using one of the above techniques
  - $$$

Buy more resources

- Large content distribution networks (e.g. Akamai, CloudFlare) can handle very large attacks
- Each attacker gets diverted (i.e., via DNS) to local CDN server instead of target
  - Total bandwidth CND can handle is the product of the bandwidth to all CND servers
  - Big CDNs have weathered attacks well in excess of 1TB/s
- Issue: who pays for that? $$$

Special case: Reflection attacks

- Spoof source address to be that of victim
- Common example
  - Send name server request to 1000s of DNS servers *on behalf* of victim (i.e., spoof victim in source address)
  - All name servers send responses to victim
- Advantages (for attacker)
  - Amplification: some protocols response size >> request size (NTP, DNS, SSDP)
  - Anonymity: attack doesn't come from attacker's machines

Solution: try not to have "open" Internet services that allow this

DoS Summary

- In general, some of the toughest problems to solve
  - Network service model allows unsolicited requests
  - Bad guys can leverage large # of resources
  - Hard to attribute network actions
  - Few systems can account for effort spent per request or isolate impact of some requests from others
- DDoS-based extortion and retribution
  (e.g., against security companies) is not uncommon
  Also widely used for political disruption

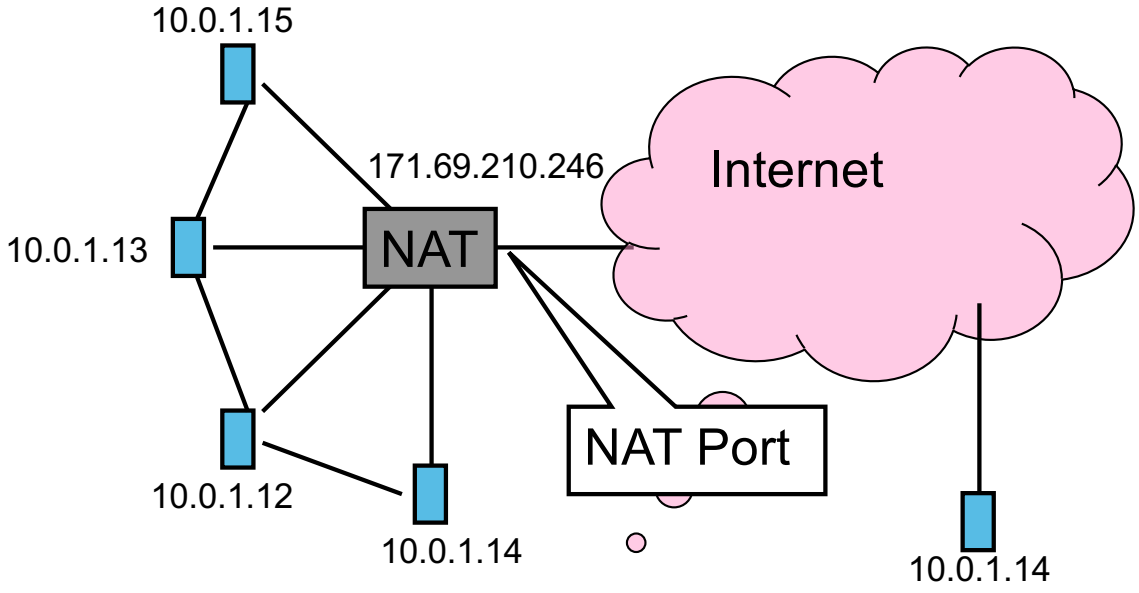
Next time

Return to Crypto I

Extra: Network Address Translation

Idea: Break the invariant that IP addresses are globally unique

- Special addresses that are only **local**: 10.x.x.x, 192.168.x.x and 172.16.0.0-172.31.255.255



Extra: Typical NAT Behavior

NAT maintains a table of the form:
<client IP> <client port> <NAT ID>

Outgoing packets (on internal port):
- Look for (client IP address, client port) in the mapping table (is there an existing mapping?)
- If not found, allocate a new unique NAT ID and replace source port with NAT ID (same size as port #)
- If found, replace client port with previously allocated NAT ID
- Replace source address with NAT address (i.e., public IP address)

Incoming Packets (on NAT port)
- Look up destination port number as NAT ID in port mapping table
- If found, replace destination address and port with client entries from the mapping table
- If not found, the packet is not for us and should be rejected

Table entries expire after some time of no activity to allow them to be garbage collected

Extra: NAT Pro/Con

Benefits

- Only allows connections to the outside that are established from *inside.*

  Hosts from outside can only contact internal hosts that appear in the mapping table, and they're only added when they establish the connection

- Don't need as large an external address space

  (e.g., 10 machines can share 1 IP address)

Costs

- Rewriting IP addresses isn't always easy (what if they appear in the **content** of the packet too? e.g., FTP. Then what happens to sequence numbers?)
- Breaks some protocols (e.g., some streaming protocols have client invoke server and then server opens a **new connection to client**)
- But we've largely paid these costs, NAT is now ubiquitous…