CSE508 Network Security

2/3/2016

Core Protocols: DNS

Michalis Polychronakis

Stony Brook University

Domain Name System

DNS maps domain names to IP addresses

- "Phonebook" for the internet
- *Client: I want to connect to:* `www.cs.stonybrook.edu`
- *DNS server: here is its IP address:* `130.245.27.2`

Distributed, hierarchical, reliable database

- Replaced the manually maintained /etc/`hosts` file

Domain names are registered and assigned by *registrars* accredited by ICANN

Not always a one-to-one mapping

- Virtual hosting: many names to a single IP address
- Load balancing/fault tolerance: single name to many addresses

DNS Server Hierarchy

Hierarchically divided name space

`.edu` ➔ `stonybrook.edu` ➔ `cs.stonybrook.edu` ➔ `www.cs.stonybrook.edu`

Root name servers

Responsible for top-level domains (TLDs): `.com`, `.edu`, `.net`, …

Point to the *authoritative name server* of each TLD ➔ managed by government or commercial organizations

```
$ curl http://data.iana.org/TLD/tlds-alpha-by-domain.txt |wc -l
1545
```

Authoritative name servers are responsible for a set of names belonging into a *zone*

Leaf nodes in the DNS hierarchy manage the zone of a single domain (e.g., `stonybrook.edu`)

DNS Resolvers

Query DNS servers and resolve the requested resource

Main query types:

Non-recursive: query a single server and receive a response

- May be a partial response

Recursive: query a single server, which may then query (as a client itself) other DNS servers on behalf of the requester

- Has to reply with the requested response or "doesn't exist" (cannot refer the client to a different DNS server)

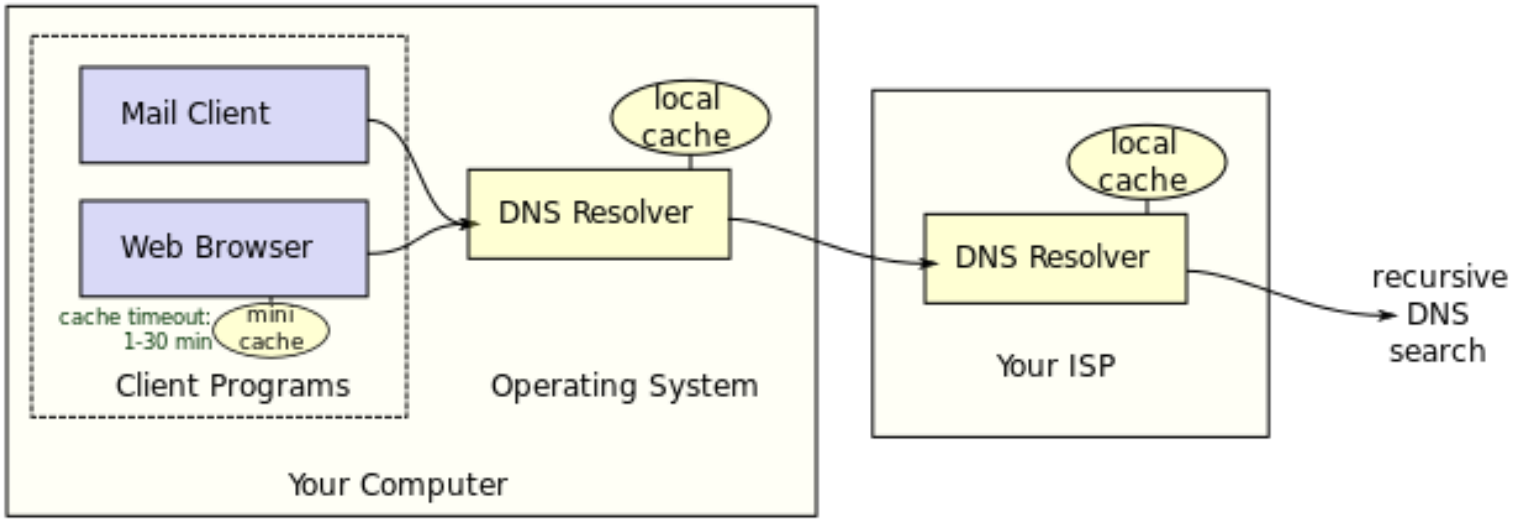
Iterative: query a chain of one or more DNS servers

- Each server returns the best answer it has
- If it doesn't have an exact match, it returns a *referral:* a pointer to an authoritative server lower in the chain

Walking the Tree: End User

User applications place resolution requests to the *stub resolver* of the OS

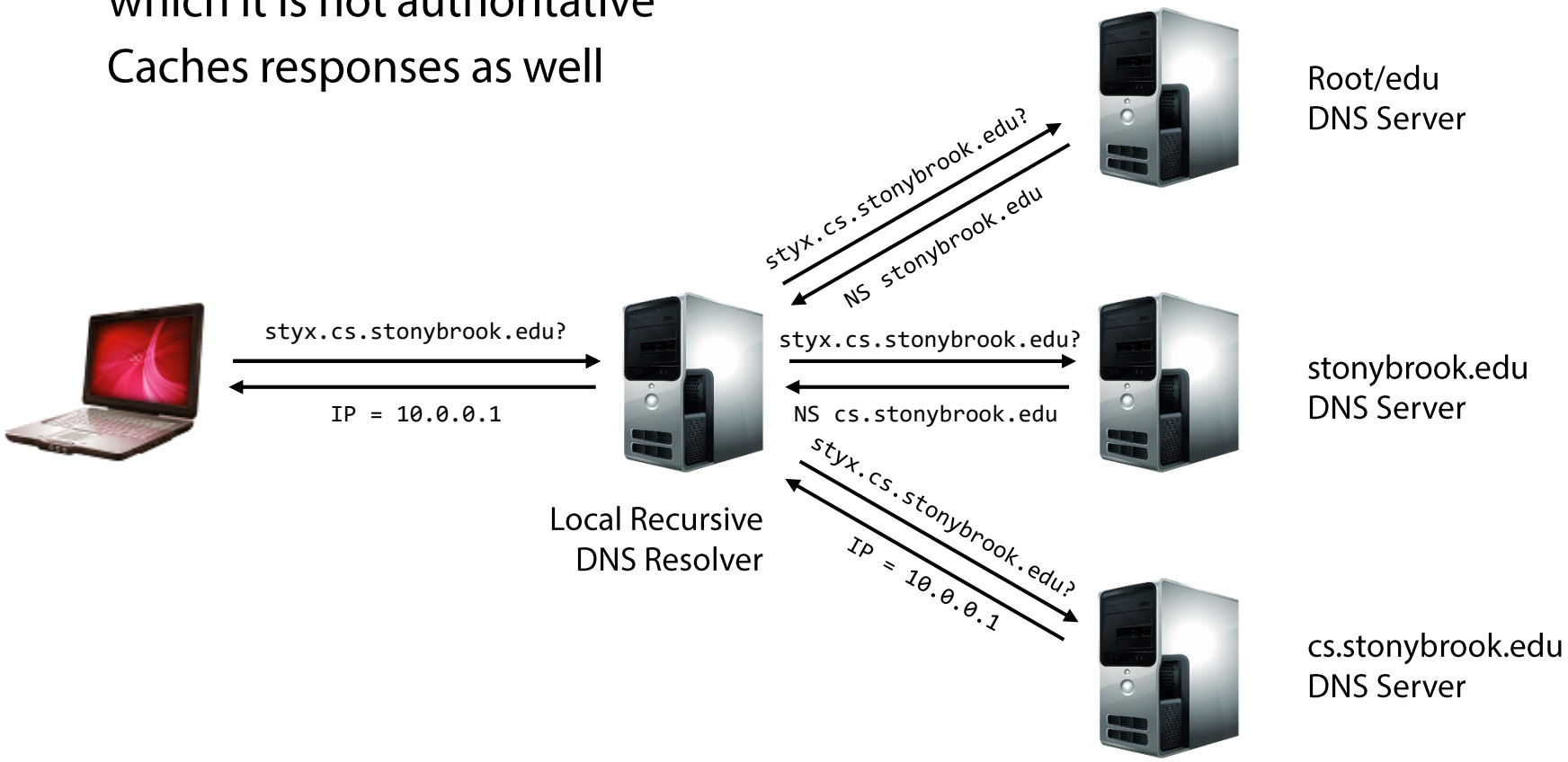
- The stub resolver then sends DNS queries to a *recursive resolver*
- Caches responses for future queries (TTL specified by owner)
- Negative responses are cached as well ➔ save time for nonexistent sites (e.g. due to misspelling)



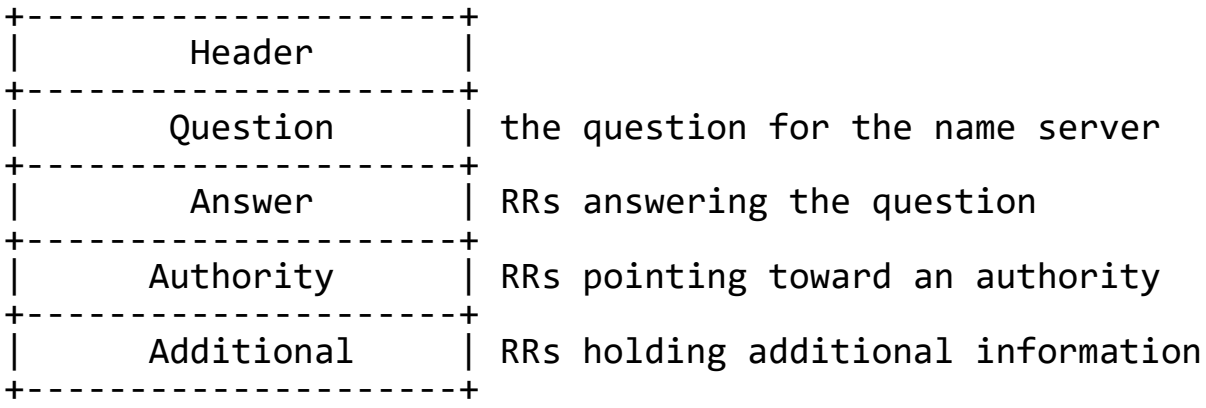
Walking the Tree: Recursive Resolver

Hosts know at least one local DNS recursive resolver

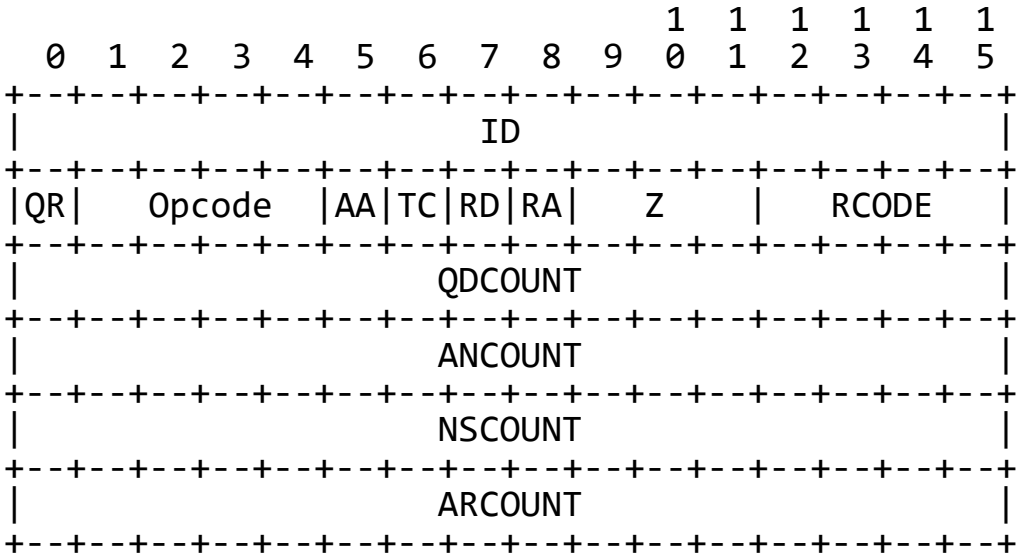
- Usually specified by the ISP or organization through DHCP – users can manually override it
- Uses the hierarchy of zones and delegations to respond to queries for which it is not authoritative
- Caches responses as well



DNS message format



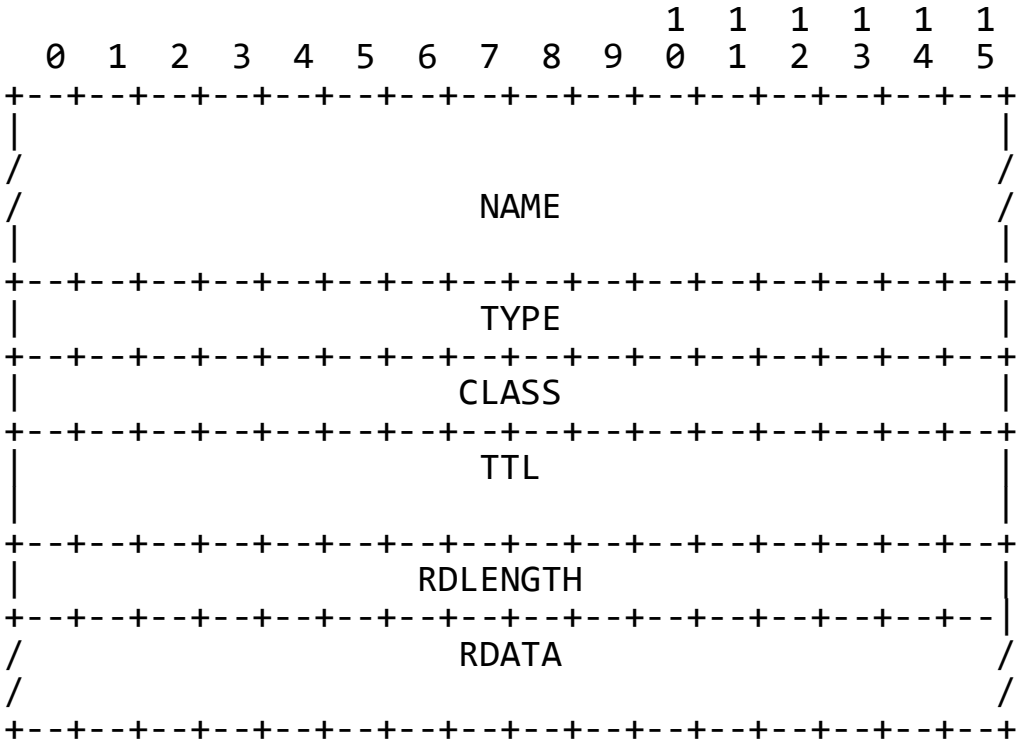
DNS header format



Primarily uses UDP for queries/responses (port 53)

TCP sometimes used for long responses and zone transfers

DNS resource record format



NAME	Name of the node to which this record pertains
TYPE	Type of RR in numeric form (e.g., 15 for MX RRs)
CLASS	Class code
TTL	Count of seconds that the RR stays valid
RDLENGTH	Length of RDATA field
RDATA	Additional RR-specific data

Types of Resource Records

Besides translating host addresses, DNS is in essence a generic "directory" for other host-related information

A: host address

NS: authoritative name server

MX: mail server of domain

CNAME: aliases for other names (not IP addresses)

PTR: map IP addresses to names (reverse lookup)

TXT: arbitrary data associated with the domain

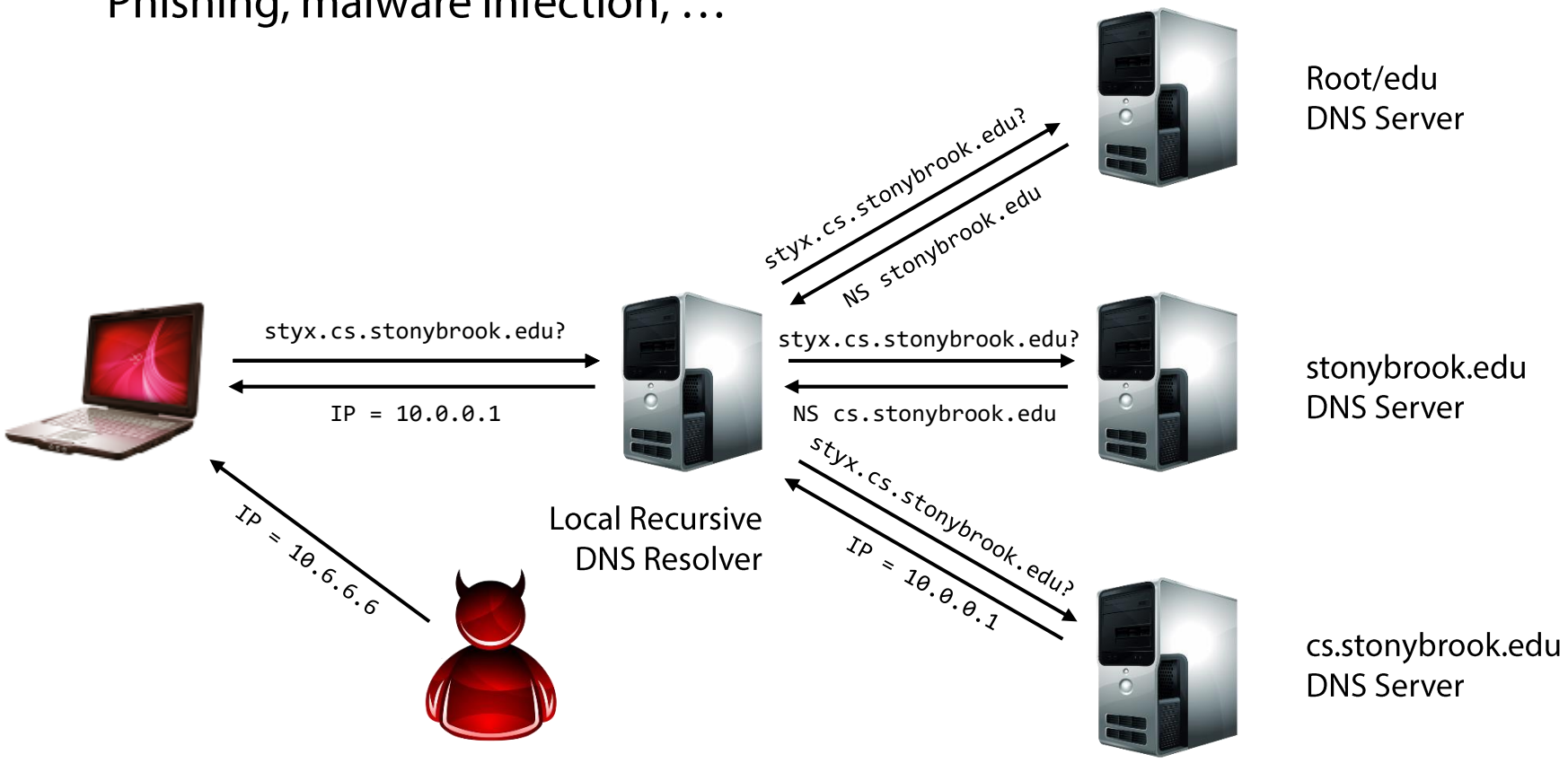
HINFO: host information

DNS Spoofing/Cache Poisoning

No authentication (reminds something?)

Responses can be spoofed!

- Point to a different address of the attacker's choosing
- Phishing, malware infection, …



Subverting Name-based Authentication

Described by Steven Bellovin (1990)

Trusted access based on host names (not a good idea)

- Reverse DNS lookup to check if client's host name is contained in a list of authorized hosts
- Example: "r-utilities" perform name-based authentication (e.g., permit all hosts in `.rhosts` to `rsh`/`rlogin` in)

Attack: fake a PTR record for an attacker-controlled IP address to return a trusted hostname

- When `rsh`/`rlogin` receives the connection, the reverse lookup using the attacker's originating IP will return a trusted name…

Fix: cross-check the returned host name by performing another lookup ➔ *forward query*

DNS Poisoning: Different Vantage Points

Off-path: attackers cannot observe any DNS queries and responses *(blind)*

- Blind packet injection: must guess the proper values in the response fields according to the query
- Race condition: forged response must arrive before the real one

On-path: attackers can passively observe the traffic (queries) and inject properly forged responses *(MotS)*

- Easy to mount in WiFi networks, by ISPs, …
- Race condition: forged response must arrive before the real one

In-path: attackers can block responses from reaching the victim, and inject forged ones instead *(MitM)*

- But then the attacker can do so much more…

DNS TXID

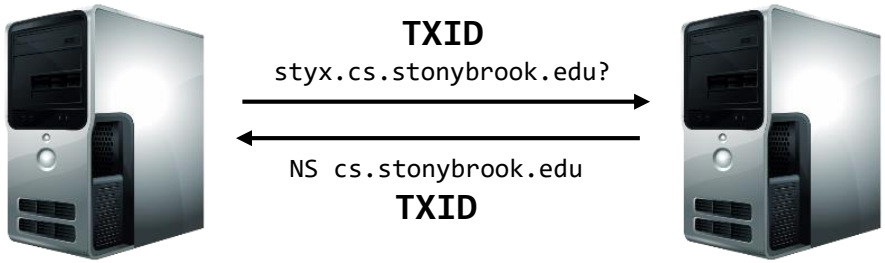
Synchronization mechanism between clients and servers

16-bit transaction identifier

- Randomly chosen for each query
- Response accepted only if TXIDs match
- Response cached according to TTL (e.g., one day)

Attacker has to win a race

- Guess the correct TXID
- Response *src IP* and *dst port* should match query *dst IP* and *src port*



It's possible!

Kaminsky Attack (Dan Kaminsky, 2008)

Goal: poison server's DNS cache entry for `example.com`

- Cannot just send random DNS packets to the recursive
- The server will only accept responses to *pending* queries

Requirements for a successful forged response:

- Matching source and destination IP address ➔ trivial
- Matching source and destination UDP port ➔ old DNS servers would use 53 for source port too (even if different, can be easily inferred)
- Matching TXID ➔ 16 bits of randomness
- Matching question section ➔ attacker targets a particular recursive server, so can trigger a query at will

Additional issue: `www.example.com` may already be in the recursive server's cache

- In that case the recursive will not ask the authoritative

Kaminsky Attack (Dan Kaminsky, 2008)

Query the recursive with any subdomain not in the cache

- Non-existent subdomains are fine: `foo1.example.com`
- Not affected by TTL (e.g., as would be the case for `www.example.com`)
- Causes the target resolver to query the authoritative server(s) for the requested subdomain

The attacker then floods the resolver with a large number of forged responses

- Each containing a different guess of the query's TXID

Fake referral →

```
;; ANSWER SECTION:
foo1.example.com.       120   IN A   10.0.0.10
;; AUTHORITY SECTION:
example.com.          86400   IN NS
ns1.example.com.
;; ADDITIONAL SECTION:
ns1.example.com.     604800   IN A   10.6.6.6
```

If the race is lost, just repeat with a different subdomain!

Kaminsky Attack: Key Insights

The recursive will always contact the authoritative for example.com for any lookup of a non-existent domain

- E.g., `foooo1.example.com`

The attacker can poison the cache with values in the additional RR field

- It's fine that the query is for a non-existent domain

Today's internet speeds allow flooding the server with thousands of packets before the real response arrives

- Likely more than enough TXID guesses

Fix: source UDP port randomization

- Orders of magnitude higher TXID + port entropy

Pharming

Mostly traffic redirection attacks at the client side

Malware can alter local DNS settings

- Change the system's (or the local router's) DNS server
- Add entries in /etc/hosts
- Example: DNSChanger: est. 4M infected computers, US$14M profit (FBI's "Operation Ghost Click")

Drive-by pharming

- A malicious web page contains JavaScript code that alters the local router's DNS server *from the inside LAN*

Dynamic pharming

- Quickly switch mapping of `bank.com` between a malicious and a real IP
- First serve malicious script, then switch to the real site ➔ same origin policy is bypassed

Other DNS Attacks

DNS hijacking by attacking registrars

- Social engineering, stolen credentials, …

DoS on root/critical servers

- Or other targets -> DNS amplification attacks

Typosquatting/registering expired domains

- Phishing – www.paypa1.com
- Hijack scripts hosted on expired domains still in use by other web pages

Covert DNS communication

- Data exfiltration, C&C, …

Zone transfers

- Reconnaissance

Server bugs

- System compromise

Censorship

PyPI Python repository hit by typosquatting sneak attack

19 SEP 2017 1

Security threats, Vulnerability

Previous: DOJ lets itself off the privacy hook

Next: Apple's new tracking protection is "sabotage", clai...

by John E Dunn

Somebody with time on their hands has tested out a devious new form of typosquatting targeting developers installing Python packages from the PyPI (Python Package Index) repository.

According to an advisory posted to the Slovak National Security Office (NBU), ten packages for Python 2.x were removed from the site after their setup.py files were found to contain malicious code. The bad code was hiding in plain site in the repository, using filenames either nearly identical to, or which could be mistaken for, legitimate ones.

Involved in numerous secret raids on club penguin, and I have

M-I-C-K-E-Y-M-C

Guest

Cybersecurity Incide

Guest

equifaxsecurity.com

Secure | https://www.equifaxsecurity2017.com

EQUIFAX

Return to equifax.com

To enroll in complimentary identity theft protection and credit file monitoring, click here.

Cybersecurity Incident & Important Consumer Information

Consumer Notice | FAQs | Potential Impact | Enroll | TrustedID Premier | Contact Us

Equifax Announces Cybersecurity Incident Involving Consumer Information

No Evidence of Unauthorized Access to Core Consumer or Commercial Credit Reporting Databases

Company to Offer Free Identity Theft Protection and Credit File Monitoring to All U.S. Consumers

September 7, 2017 — Equifax Inc. (NYSE: EFX) today announced a cybersecurity incident potentially impacting approximately 143 million U.S. consumers. Criminals exploited a U.S. website application vulnerability to gain access to certain files. Based on the company's investigation, the unauthorized access occurred from mid-May through July 2017. The company has found no evidence of unauthorized activity on Equifax's core consumer or commercial credit reporting databases.

The information accessed primarily includes names, Social Security numbers, birth dates, addresses and, in some instances, driver's license numbers. In addition, credit card numbers for approximately 209,000 U.S. consumers, and certain dispute documents with personal identifying information for approximately 182,000 U.S. consumers, were accessed. As part of its

DNSSEC

Goal: enable authentication and ensure the integrity of DNS requests and responses

- Non-goals: availability, confidentiality

Cryptographically signed resource records

- Resolvers can verify the signature

Two new resource types:

DNSKEY: creates a hierarchy of trust within each zone

- Name = Zone domain name
- Value = Public key for the zone

RRSIG: Prevents hijacking and spoofing

- Name = (type, name) tuple, i.e. the query itself
- Value = Cryptographic signature of the query results

Not a complete solution

- Enables DoS amplification/CPU exhaustion attacks
- Forgery of delegation records still possible
- No "last mile" protection