Information Security CS 526

Topic 19: DNS Security

Domain Name System

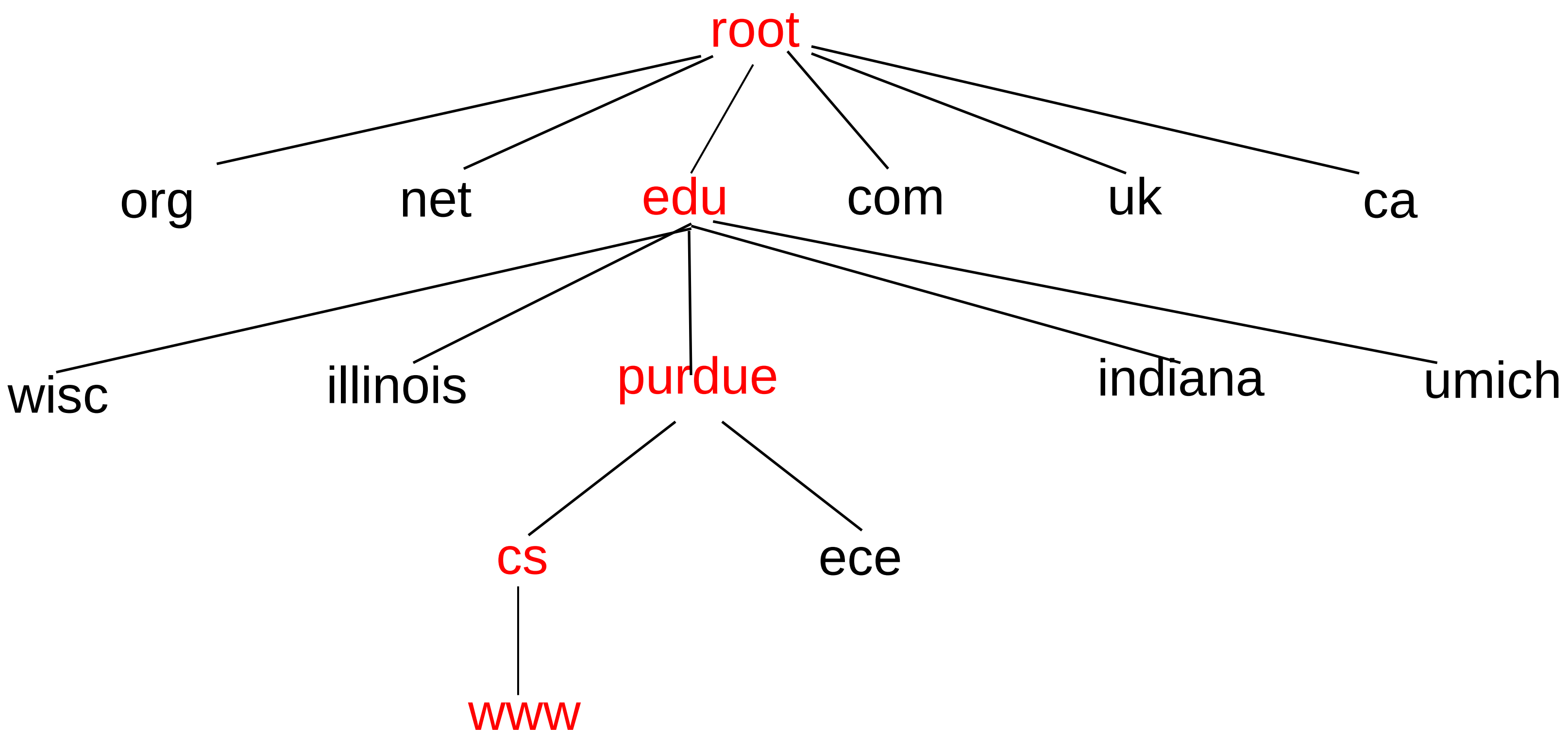
- Translate host names to IP addresses
  - E.g., www.xyz.com ➔ 74.125.91.103
  - Why is needed?
    - E.g. akami.
- And back
  - From IP addresses to DNS name

DNS is a Distributed Database

- Information is stored in a distributed way
- Highly dynamic
- Decentralized authority

Domain Name System

- Hierarchical Name Space



Domain Name System

Domain Name Servers

- Top-level domain (TLD) servers:
  - responsible for com, org, net, edu, etc, and all top-level country domains, e.g. uk, fr, ca, jp.
  - Network Solutions maintains servers for “.com”
- Authoritative DNS servers:
  - organization’s DNS servers, providing authoritative hostname to IP mappings for organization’s servers.
  - can be maintained by organization or service provider.

Domain Name Servers - 2

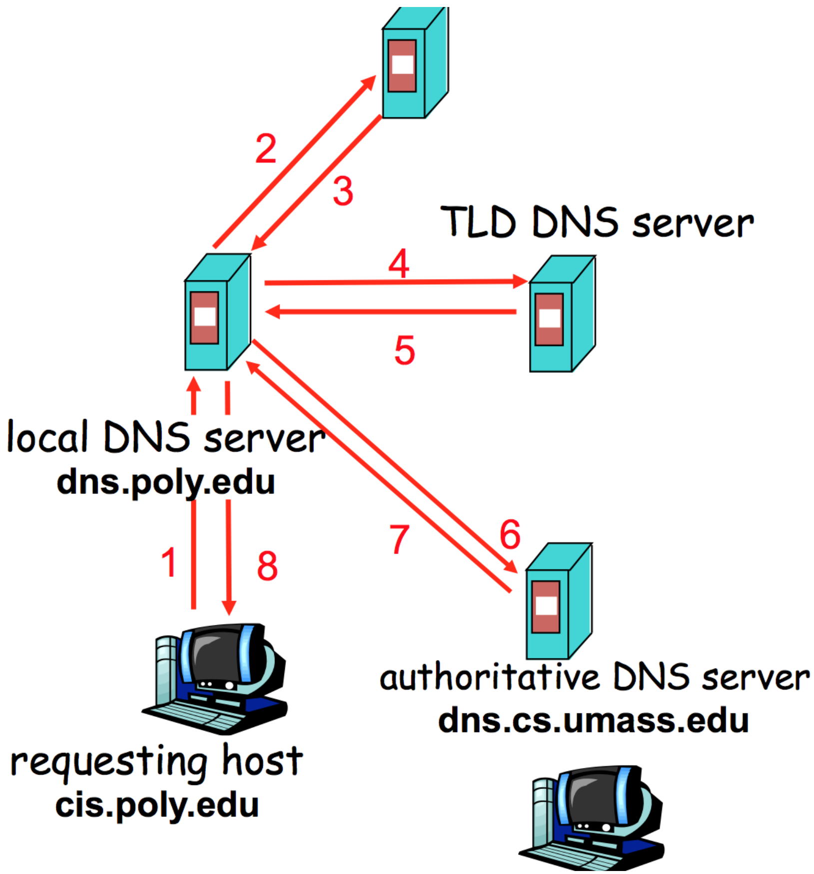
- Local Name Server
  - does not strictly belong to hierarchy
  - each ISP (residential ISP, company, university) has one.

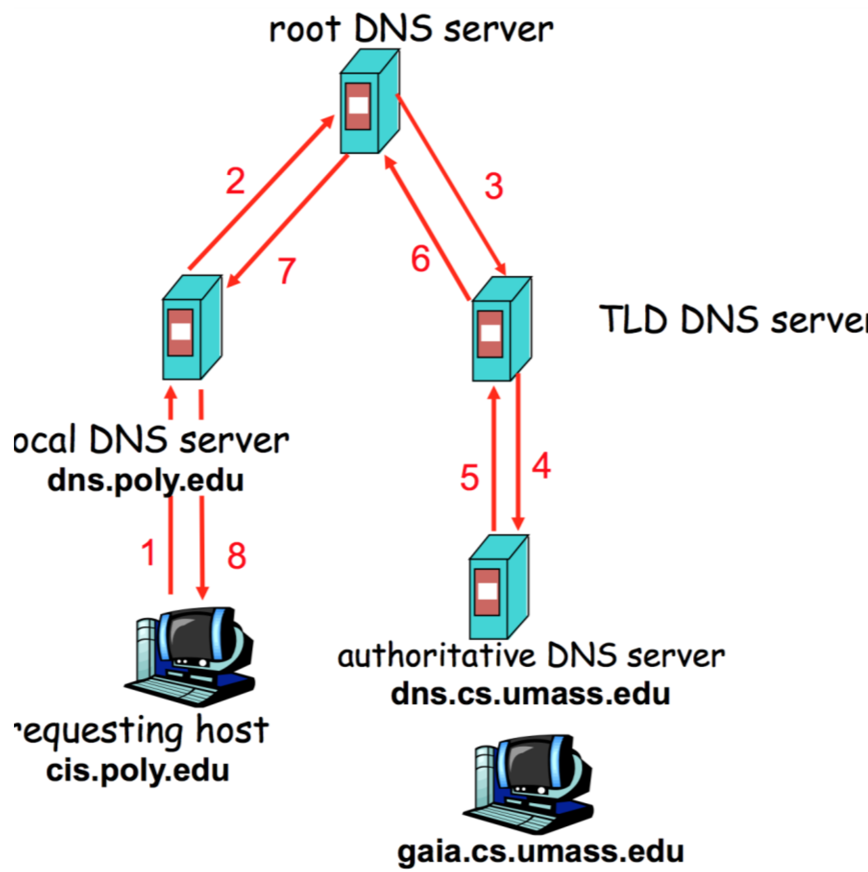
DNS Resolving

- When host makes DNS query, query is sent to its local DNS server.
  - acts as proxy, forwards query into hierarchy.

- Two resolving schemes:
  - Iterative, and
  - Recursive.

DNS Resolving - 2

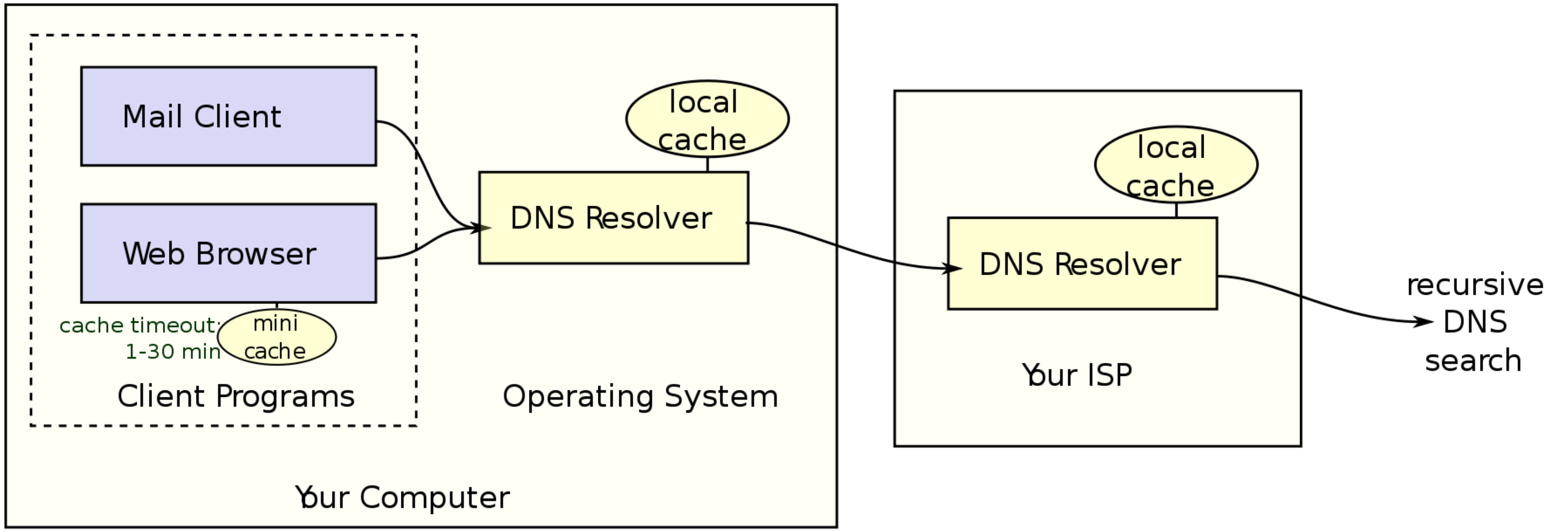




Caching

- DNS responses are cached
  - Quick response for repeated translations
- Negative results are also cached
  - Save time for nonexistent sites, e.g. misspelling
- Cached data periodically times out
  - Each record has a TTL field

Caching - 2



Mail Client
Web Browser
cache timeout:
1-30 min
mini
cache
Client Programs
local
cache
DNS Resolver
Operating System
Your Computer
local
cache
DNS Resolver
Your ISP
recursive
DNS
search

Inherent DNS Vulnerabilities

- Users/hosts typically trust the host-address mapping provided by DNS
  - What bad things can happen with wrong DNS info?
- DNS resolvers trust responses received after sending out queries.
  - How to attack?
- Obvious problem
  - No authentication for DNS responses

User Side Attack - Pharming

- Exploit DNS poisoning attack
  - Change IP addresses to redirect URLs to fraudulent sites
  - Potentially more dangerous than phishing attacks
    - Why?

- DNS poisoning attacks have occurred:
  - 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

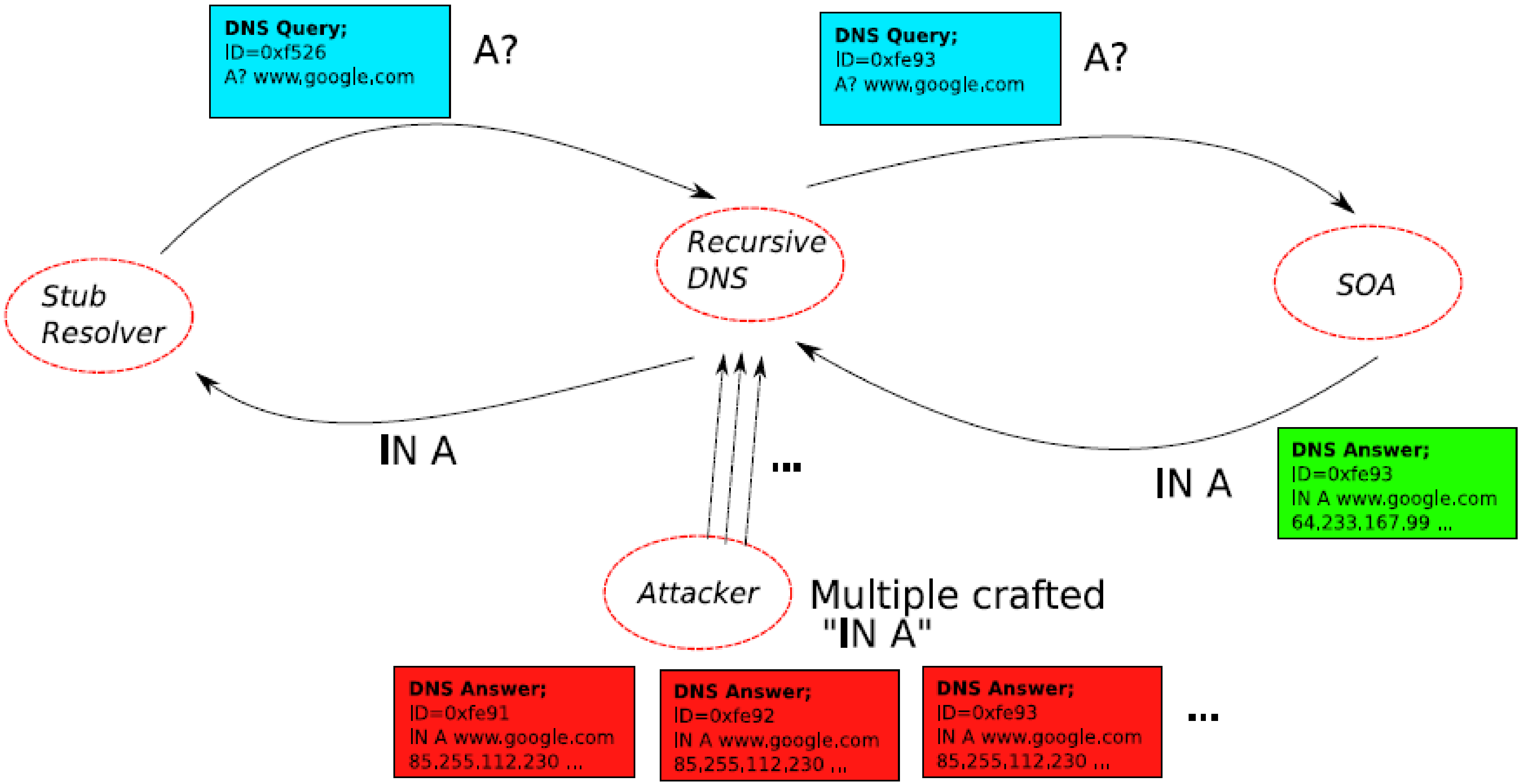
DNS Cache Poisoning

- Attacker wants his IP address returned for a DNS query
- When the resolver asks ns1.google.com for www.google.com, the attacker could reply first, with his own IP
- What is supposed to prevent this?
- Transaction ID
  - 16-bit random number
  - The real server knows the number, because it was contained in the query
  - The attacker has to guess

DNS cache poisoning - 2

- Responding before the real nameserver
  - An attacker can guess when a DNS cache entry times out and a query has been sent, and provide a fake response.
  - The fake response will be accepted only when its 16-bit transaction ID matches the query
  - CERT reported in 1997 that BIND uses sequential transaction ID and is easily predicted
    - fixed by using random transaction IDs

DNS cache poisoning: Racing to Respond First



DNS cache poisoning (Schuba and Spafford in 1993)

- DNS resource records (see RFC 1034)
  - An “A” record supplies a host IP address
  - A “NS” record supplies name server for domain
- First, guess query ID:
  - Ask (dns.target.com) for www.evil.org
  - Request is sent to dns.evil.org (get quid).
- Second, attack:
  - Ask (dns.target.com) for www.yahoo.com
  - Give responses from “dns.yahoo.com” to our chosen IP.

Defense Using The Bailiwicks Rules

- The bailiwick system prevents foo.com from declaring anything about “com”, or some other new TLD, or www.google.com

- Using the bailiwicks rules
  - The root servers can return any record
  - The com servers can return any record for com
  - The google.com servers can return any record for google.com

DNS cache poisoning – Birthday attack

- Improve the chance of responding before the real nameserver (discovered by Vagner Sacramento in 2002)
  - Have many (say hundreds of) clients send the same DNS request to the name server
    - Each generates a query
  - Send hundreds of reply with random transaction IDs at the same time
  - Due to the Birthday Paradox, the success probability can be close to 1
    - 300 will give you 50%.
    - 700 will give you 1.07%

DNS poisoning – So far

- Early versions of DNS servers deterministically incremented the ID field
- Vulnerabilities were discovered in the random ID generation
  - Weak random number generator
  - The attacker is able to predict the ID if knowing several IDs in previous transactions
- Birthday attack
  - 16- bit (only 65,536 options).
  - Force the resolver to send many identical queries, with different IDs, at the same time
  - Increase the probability of making a correct guess

DNS cache poisoning - Kaminsky

- Kaminsky Attack
  - Big security news in summer of 2008
  - DNS servers worldwide were quickly patched to defend against the attack

- In previous attacks, when the attacker loses the race, the record is cached, with a TTL.
  - Before TTL expires, no attack can be carried out
  - Posining address for google.com in a DNS server is not easy.

What is New in the Kaminsky Attack?

- The bad guy does not need to wait to try again
- The bad guy asks the resolver to look up www.google.com
  - If the bad guy lost the race, the other race for www.google.com will be suppressed by the TTL
- If the bad guy asks the resolver to look up 1.google.com, 2.google.com, 3.google.com, and so on
  - Each new query starts a new race
- Eventually, the bad guy will win
  - he is able to spoof 183.google.com
  - So what? No one wants to visit 183.google.com

Kaminsky-Style Poisoning

- A bad guy who wins the race for “183.google.com” can end up stealing “www.google.com” as well

- Original malicious response:
  - google.com NS www.google.com
  - www.google.com A 6.6.6.6
- Killer response:
  - google.com NS ns.badguy.com

Kaminsky-Style Poisoning (cont')

- Why it succeeded:
  - Can start anytime; no waiting for old good cached entries to expire
  - No "wait penalty" for racing failure
  - The attack is only bandwidth limited
- Defense (alleviate, but not solve the problem)
  - Also randomize the UDP used to send the DNS query, the attacker has to guess that port correctly as well (increase the space of possible IDs).

DNS Poisoning Defenses

- Difficulty to change the protocol
  - Protocol stability (embedded devices)
  - Backward compatibility.
- Long-term
  - Cryptographic protections
    - E.g., DNSSEC, DNSCurve
  - Require changes to both recursive and authority servers
  - A multi-year process
- Short-term
  - Only change the recursive server (local DNS).
  - Easy to adopt

Short-Term Defenses

- Source port randomization
  - Add up to 16 bits of entropy
  - NAT could de-randomize the port
- DNS 0x20 encoding
  - From Georgia tech, CCS 2008
- Tighter logic for accepting responses

DNS-0x20 Bit Encoding

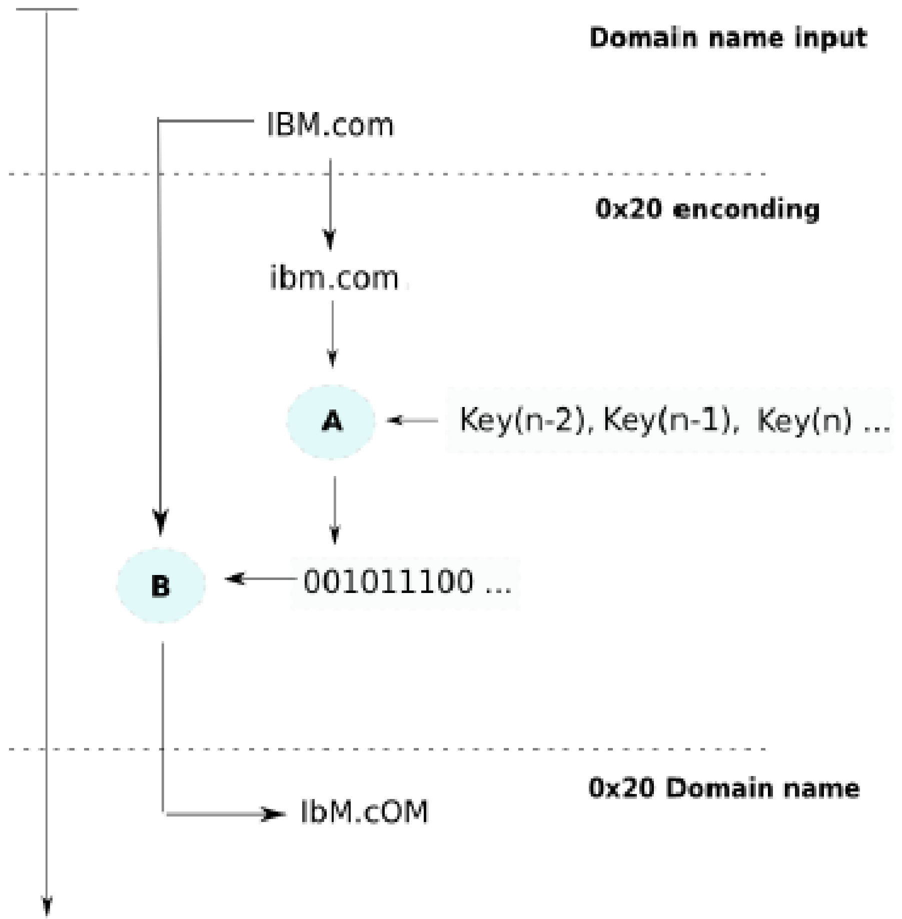
- DNS labels are case insensitive
- Matching and resolution is entirely case insensitive
- A resolver can query in any case pattern
  - E.g., WwW.ExAmpLe.cOM
  - It will get the answer for www.example.com

DNS-0x20 DNS Encoding (cont')

- A DNS response contains the query being asked
- When generating the response, the query is copied from the request exactly into the response
  - The case pattern of the query is preserved in the response
- Open source implementations exhibit this behavior
  - The DNS request is rewritten in place
- The mixed pattern of upper and lower case letters constitutes a channel, which can be used to improve DNS security
  - Only the real server knows the correct pattern

Query Encoding

- Transforms the query into all lowercase
- Encrypt the query with a key shared by all queries on the recursive server (A)
- The cipher text is used to encode the query
  - 0: buff[i] |= 0x20 (upper)
  - 1: buff[i] &= 0x20 (lower)



DNS-0x20 Encoding Analysis

- Do existing authority servers preserve the case pattern?
  - Scan 75 million name servers, 7 million domains
- Only 0.3% mismatch observed

Type	Mismatch	Mismatch pct.	Domain scanned
.com TLD	15451	0.327%	4786993
.net TLD	4437	0.204%	2168352

DNS-0x20 Encoding Analysis (cont')

- Not every character is 0x20 capable
- Improve the forgery resistance of DNS messages only in proportion to the number of upper or lower case characters
  - cia.gov 6-bit entropy
  - licensing.disney.com 18-bit entropy
  - 163.com 3-bit entropy
- TLDs are also vulnerable to Kaminsky-style attacks; but they have few 0x20-capable bits

Other DNS attacks

- Attacking home routers/gateways
- Incidence in Mexica in 2008
  - an email sent to users
  - email include URL (HTTP requests) to the HTTP-based interface of wireless routers
  - using the default password to reconfigure the router/gateway

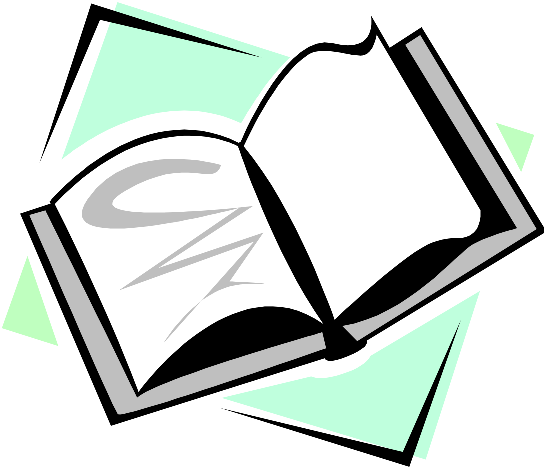
Long Term Solution

- DNSSEC:
  - Authenticate responses.
  - Google DNS now is enabled by default.

- Challenges in deployment:
  - Response is large, might no linger fit in single UDP message.
  - Legacy software and machines.

Readings for This Lecture

- Optional:
  - First attack by Schuba and Spafford - http://www.openbsd.org/advisories/sni_12_resolverid.txt
  - **An Illustrated Guide to the Kaminsky DNS Vulnerability**
  - Dan Kaminsky's Black Hat presentation (PowerPoint)



Coming Attractions …

- Non-interference and non-deducability