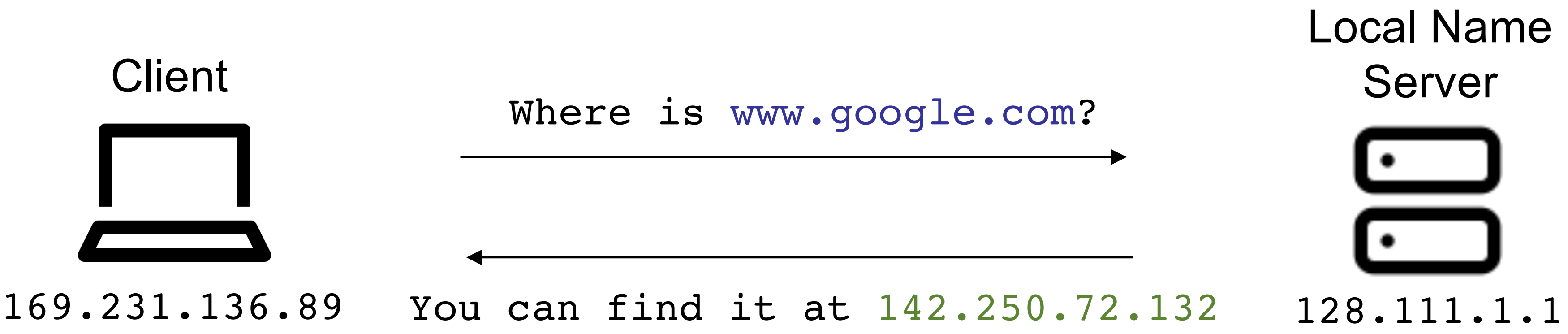
CS 177 - Computer Security

DNS Security

Naming

- Network identities (names) are integral to addressing, routing, and access control
- On the Internet, names are IP addresses
- Raw IP addresses are not always the right abstraction
- Alternatively, we can use domain names
- Domain names are memorable labels (such as www.google.com) that map to an IP address (142.250.72.132)

Name Resolution



Client

169.231.136.89

`Where is www.google.com?`

`You can find it at 142.250.72.132`

Local Name Server

128.111.1.1

How does the client find the local name server?

How to map names to IP addresses?

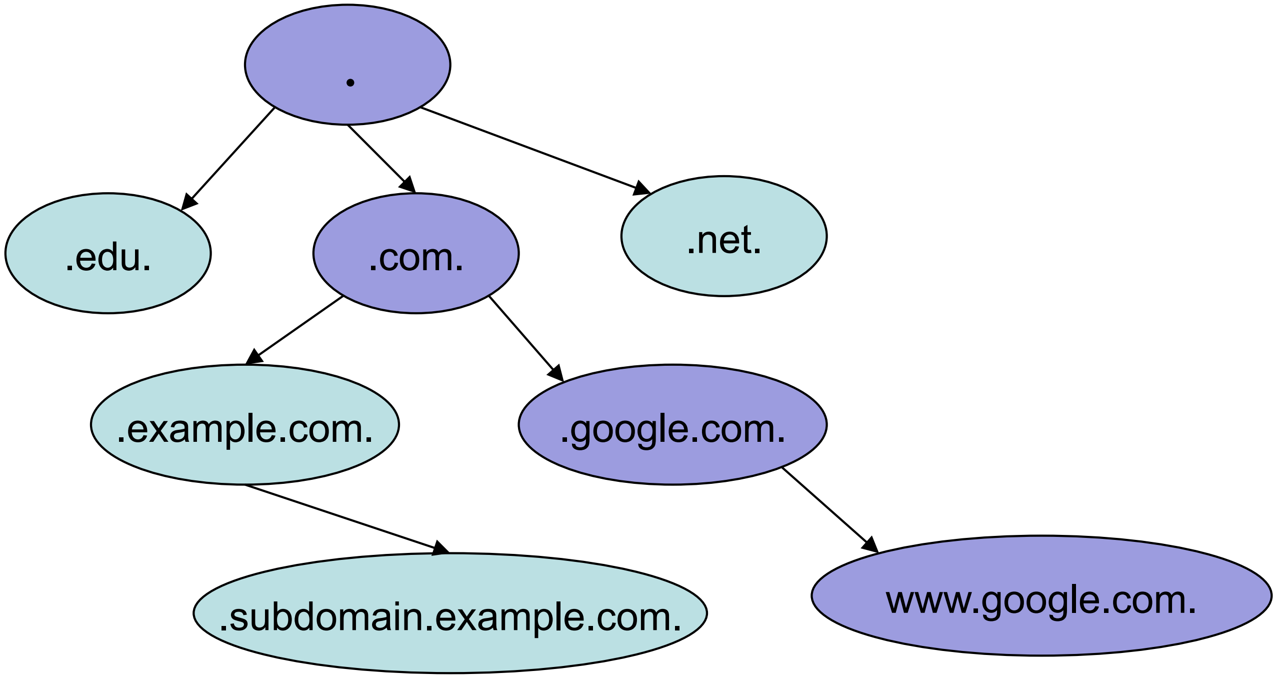
- Initially, there was a big file that everyone had to download
- Today, we use a distributed service called the Domain Name System (DNS)
- But we can still store some mappings locally (in a file)

Domain Name System (DNS)

Domain Name Service (DNS)

- Initially specified in RFC 1034/1035
- Distributed database that maps names into IP addresses (and vice versa)
- Name space is hierarchically divided in domains
- Each domain is managed by a name server
- Clients access name server resolution services through a resolver library
- DNS uses mostly UDP
- Sometimes TCP for long queries and zone transfers

DNS Hierarchy



Name Server

- Name servers are responsible for mapping names of a domain
  - each name server is responsible for a part of the name space (zone)
  - this name server is called the authoritative server for that zone
  - parts of the name space can be delegated to one or more sub-zones

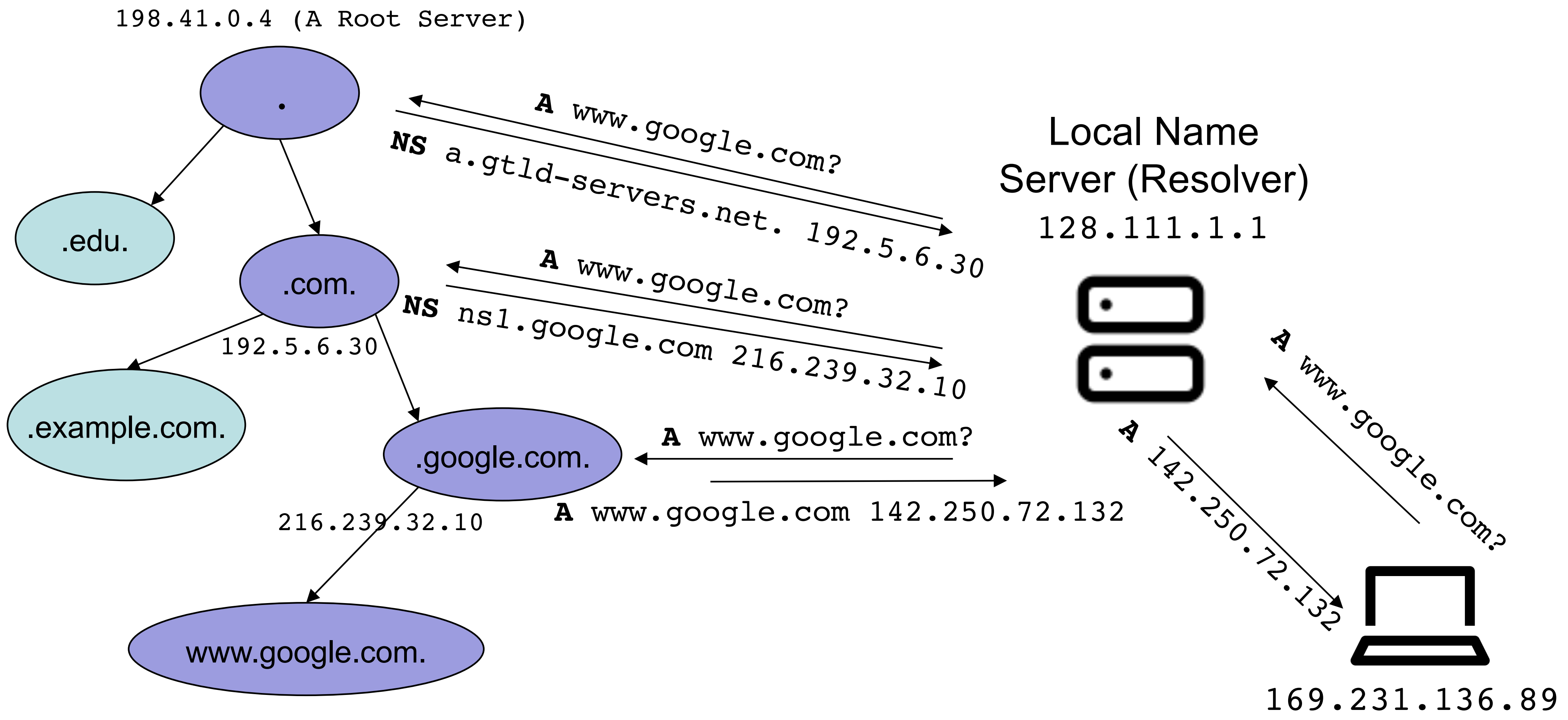
- Root name servers
  - 13 (sets of) machines distributed around the world
  - associated with the top level (root) of the DNS hierarchy
  - they have hardcoded IP addresses

Address Resolution

- A server that cannot answer a query “walks” the DNS hierarchy
  - The search starts at the top (with the root server)
  - Then, the search is following the correct branch in the hierarchy down to the authoritative server

- The results are usually maintained in a local cache

Address Resolution



DNS Clients

- At least one name server has to be specified
  - e.g., Linux uses `/etc/resolv.conf`
- Queries can be
  - recursive
    - require a name server to find the answer to the query itself
  - iterative
    - instead of the resolved name another server‘s address is returned, which can be asked
- Lookup can be performed with
  - `dig, nslookup, host`

DNS Caching

- DNS responses are cached
  - Quick response for repeated translations
  - Useful for finding nameservers as well as IP addresses (NS records for domains)
- DNS negative queries are also cached
  - Saves time for nonexistent sites, e.g., misspelled names
- 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 Packet

<table>
<tr><td>ID number (2 bytes)</td><td>Flags (2 bytes)</td></tr>
<tr><td>Question count (2)</td><td>Answer count (2)</td></tr>
<tr><td>Authority count (2)</td><td>Additional count (2)</td></tr>
<tr><td colspan="2">Question Records</td></tr>
<tr><td colspan="2">Answer Records</td></tr>
<tr><td colspan="2">Authority Records</td></tr>
<tr><td colspan="2">Additional Records</td></tr>
</table>

- ID number (16 bits): Used to match queries with responses; also called Transaction ID
- (Some) Flags
  - **`bool`** `query_or_response`
  - **`u4`** `opcode (request type)`
  - **`bool`** `authoritative_answer`
  - **`bool`** `response_truncated`
  - **`bool`** `recursion_desired`
  - **`bool`** `recursion_available`
- Counts: The number of records of each type in the DNS payload

DNS Packet

<table>
<tr><td>ID number (2 bytes)</td><td>Flags (2 bytes)</td></tr>
<tr><td>Question count (2)</td><td>Answer count (2)</td></tr>
<tr><td>Authority count (2)</td><td>Additional count (2)</td></tr>
<tr><td colspan="2">Question Records</td></tr>
<tr><td colspan="2">Answer Records</td></tr>
<tr><td colspan="2">Authority Records</td></tr>
<tr><td colspan="2">Additional Records</td></tr>
</table>

- DNS payload contains a variable number of resource records (RRs)
- Each RR is a name-value pair
- Each record is a name-value pair with a type
  - **A (answer) type records**: Maps a domain name to an IPv4 address
  - **NS (name server) type records**: Designates another DNS server to handle a domain
- Each record also contains some metadata
  - **Time to live (TTL):** How long the record can be cached

DNS Resource Records

- **Question section**: What is being asked
  - Included in both requests and responses
  - Usually, A type record with the **domain name** being looked up

- **Answer section**: A direct response to the question
  - Empty in requests
  - Used if the name server responds with the answer
  - Usually, A type record with the **IP address** of the domain being looked up

- **Authority section**: A delegation of authority for the question
  - Empty in requests
  - Used to direct the resolver to the next name server
  - Usually, **NS type record** with the zone and domain of the child name server

DNS Resource Records

- **Additional section**: Additional information to help with the response, sometimes called *glue records*
  - Empty in requests
  - Answers to questions you didn't ask
  - Usually, A type record with the domain and IP address of the child name server (since the NS record provides the child name server as a domain)

- But there is a potential security problem
  - What if the attacker runs a DNS server for `evil.com`
  - When it gets asked for the IP address for `host.evil.com`, it puts some extra information into the "additional section"

    "You can find `paypal.com` at `4.3.2.1`" (address the attacker controls)

Bailiwick Checking

- Bailiwick [Wikipedia]: *A bailiwick is usually the area of jurisdiction of a bailiff, and once also applied to territories in which a privately appointed bailiff exercised the sheriff's functions under a royal or imperial writ.*

- More general, a bailiwick is one's sphere of operations or particular area of interest

- For DNS: The resolver only accepts records if they are in the name server's zone
  - Example: The **ucsb.edu** name server can provide a record for **cs.ucsb.edu**, but not for **berkeley.edu**
  - Example: The **.edu** name server can provide an additional record for **ucsb.edu** and **berkeley.edu**, but not **google.com**

DNS Security Issues

- DNS provides rich information
  - IP addresses
  - INFO records
  - can be gathered via exhaustive queries or via zone transfers
  - IP scanning is not necessary in many cases (important with IPv6)
- DNS Hijacking
- DNS Cache Poisoning
- Kaminsky Attack

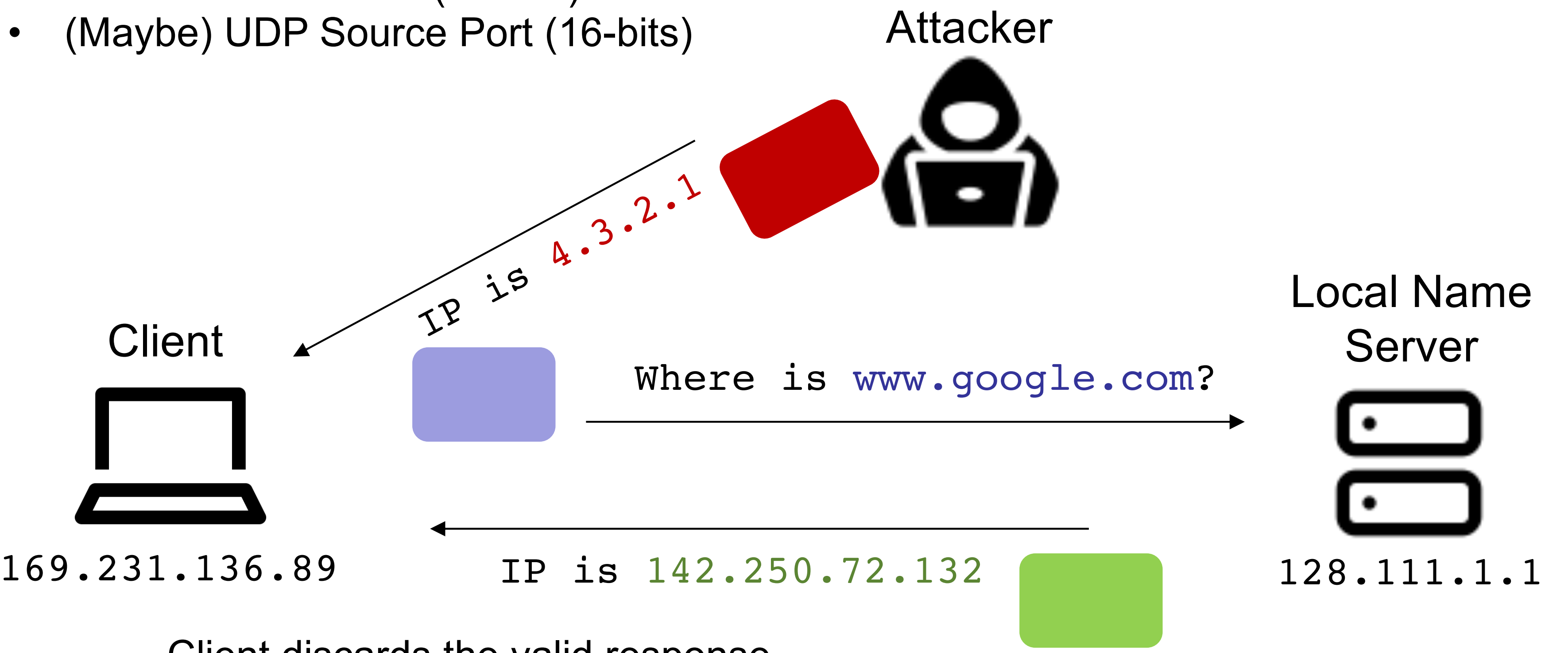
DNS Hijacking

- Relies on the fact the UDP is used
- Respond to a DNS request with incorrect data
- Respond faster than legitimate server
- It is possible to perform DNS hijacking by
  - racing with the server with respect to a client
  - racing with a server with respect to another server
- Very easy to do when the attacker sees the request
- „Blind“ DNS hijacking
  - more difficult (but how much more difficult?)

DNS Hijacking

What does the attacker need to know to be able to forge a DNS response that the client accepts?

- DNS Transaction ID (16-bits)
- (Maybe) UDP Source Port (16-bits)



Client discards the valid response

DNS Cache Poisoning

- When attacker manages to hijack a DNS request and supply a forged response, then the client will cache the incorrect (malicious) mapping (until TTL expires)

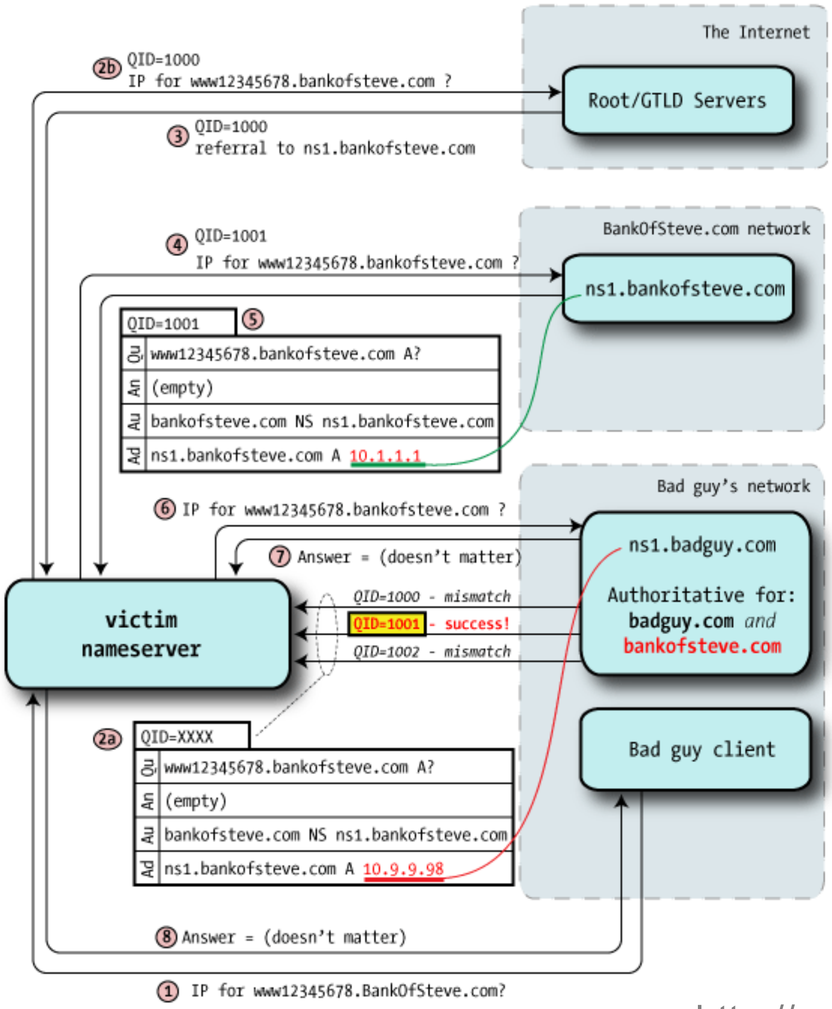
- If the attacker manages to compromise a DNS server (such as a local resolver), it will return the malicious mapping to *every* client that queries it!

Kaminsky Attack [2008]

- Remember the additional sections and the bailiwick checks?
- If the attacker wins the race and succeeds in a DNS hijacking race for a query (for example, for **server1.google.com**), they can supply additional records for anything under **google.com** (including a new name server)
- That means, if you win once, you control the complete domain

- But you only have one chance to win a race for each individual name, as the client will cache the result for that name
- What if you can trick the client into asking for many (non-existent) names? Then, you will have many chances. And you only need to win once

Kaminsky Attack [2008]



QID=1001	
Qu	www12345678.bankofsteve.com A?
An	(empty)
Au	bankofsteve.com NS ns1.bankofsteve.com
Ad	ns1.bankofsteve.com A 10.1.1.1

QID=XXXX	
Qu	www12345678.bankofsteve.com A?
An	(empty)
Au	bankofsteve.com NS ns1.bankofsteve.com
Ad	ns1.bankofsteve.com A 10.9.9.98

http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html

Defenses

- Increase search space for attacker
  - Randomize UDP source port and make sure it matches
  - Ox20 encoding: Randomly vary capitalization of name in request (DNS is case insensitive), and check that you get the same capitalization back in DNS reply
- Authenticated requests/responses
  - DNSSEC

DNSSEC

DNSSEC (DNS Security Extensions)

- Extension of the DNS protocol that ensures integrity of DNS results
    1. Origin authentication
    2. Data integrity
    3. Authenticated denial of existence

- DNSSEC does not provide
    1. Data confidentiality
    2. Service availability

- DNSSEC is backwards-compatible

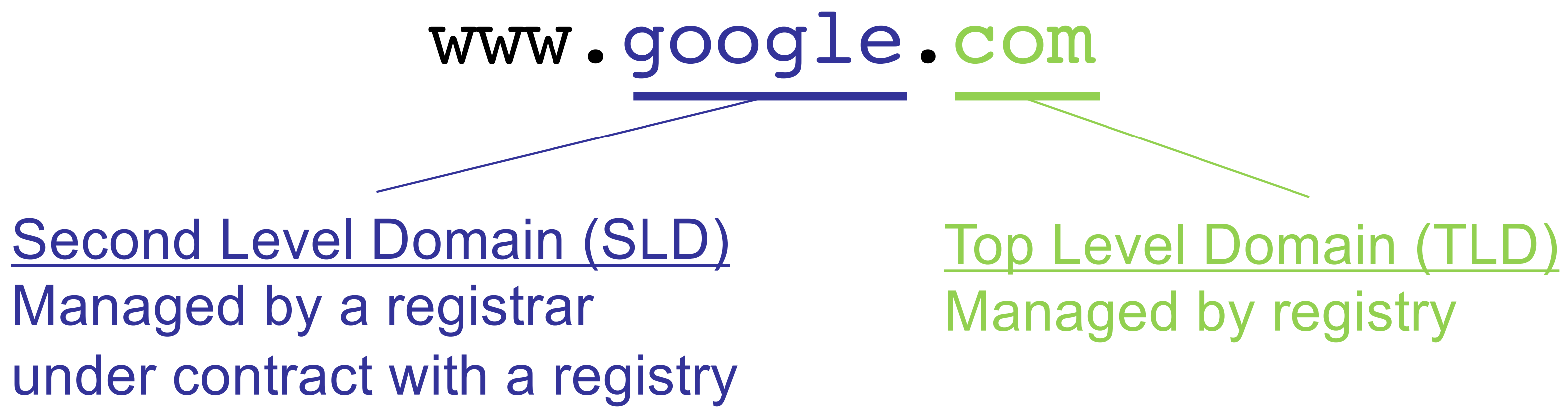
DNSSEC

- Uses a hierarchical, distributed trust system to validate records
  - similar in spirit to the certificates used for HTTPS

- Specifically, DNSSEC uses public-key cryptography to establish a chain of trust from the DNS root zone to authoritative nameservers
  - DNSSEC-aware resolvers obtain root zone public keys (trust anchors) through an out-of-band channel

- A parent server that is higher in the DNS hierarchy and that points to an authoritative name server lower in the hierarchy (child) includes public keys that allow the client to check the responses of that (child) server

Domain Name Registration



www.google.com

Second Level Domain (SLD)
Managed by a registrar
under contract with a registry

Top Level Domain (TLD)
Managed by registry

- ICANN/IANA manages the root zone
- Registries manage TLDs (e.g., VeriSign → .com)
- Registrars manage SLDs (e.g., MarkMonitor → google.com)

Registration Information

- whois database
  - query and response protocol that is widely used for querying databases that store the registered users or assignees of an Internet resource

```
> whois google.com

Domain Name: google.com
Registry Domain ID: 2138514_DOMAIN_COM-VRSN
Updated Date: 2019-09-09T15:39:04+0000
Creation Date: 1997-09-15T07:00:00+0000
Registrar Registration Expiration Date: 2028-09-13T07:00:00+0000
Registrar: MarkMonitor, Inc.
[...]
Registrant Organization: Google LLC
Registrant State/Province: CA
Registrant Country: US
[...]
Name Server: ns1.google.com
```

Domain Management

- Domains have value, and are subject to attacks
  - Trademarks, ad revenue, phishing, general abuse of trust
  - For example, Google owns goog1e.com for a reason

- **Domain squatting**: Purchasing a domain with the intent to profit of another entity's reputation

- **Domain drop catching**: Instant re-registration of expiring domains