Network Security

Zane Ma
Slides from Kirill Levchenko
University of Illinois
ECE 422/CS 461 – Fa 2019

TCP Connection Hijack

- Can we impersonate another host in *existing* TCP connection?
  - E.g. to insert data into the data stream?

TCP Connection Hijack

- Review: IP spoofing – why does it have limited utility?
  - Most protocols require back-and-forth exchange of messages
- Review: TCP 3-way handshake to establish connection
  - Exchanges certain state information: src/dst address, port, seq numbers, ack numbers, etc.
- Review: TCP sliding window
  - TCP tolerates out-of-order delivery, re-orders received packets

TCP Connection Hijack

- Can we impersonate another host in *existing* TCP connection?
  - E.g. to insert data into the data stream?
- Say we want to impersonate *A* to *B* in existing connection
- Need to know port numbers (16 bits)
  - Initiator's port number usually chosen random by OS
  - Responder's port number may be well-known port of service
- B will accept sequence numbers inside window
- B will accept ack. numbers in correct half of 32-bit seq. space
- $W \times 2^{-(16+32+1)}$ (where W is window size) chance to guess right
  - Maximum value of W is 2^{16}
- Protocol must tolerate misaligned data for attack to work

TCP Hijack Defenses

- Limit range of acceptable acknowledgement numbers up to maximum window size behind last sent data
- About $W^2 \times 2^{-(16+32+32)}$ chance to guess right
  - Maximum value of W is 2^{16}
- Can also introduce application protocol checks

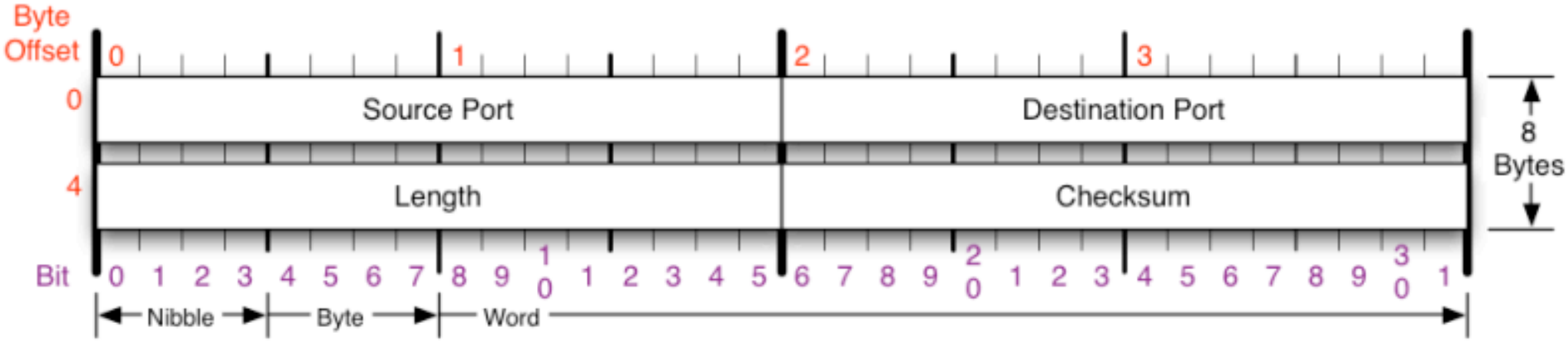
TCP Security Properties

	Passive	Off-Path	MitM
Availability	—	✗	✗
Confidentiality	✗	—	✗
Integrity	—	—	✗
Authenticity	—		✗

- TCP (with modern defenses) reduces success probability of off-path attacks to acceptable levels
  - About 2^{-48} for hijack, 2^{-32} for spoofing
- Initiator port number and all sequence numbers must be chosen uniformly at random!

User Datagram Protocol

- Sometimes we *do* only want best-effort delivery
- **User Datagram Protocol (UDP)** is a transport layer protocol that is essentially a wrapper around IP
  - Adds ports to demultiplex traffic by applications
- Checksum similar to TCP
    - Covers IP pseudo-header, UDP header, and data



Domain Name System

- Application-layer protocols (and people) usually refer to Internet host by *host name*
- Host names organized into hierarchy

www.illinois.edu

Domain Name System

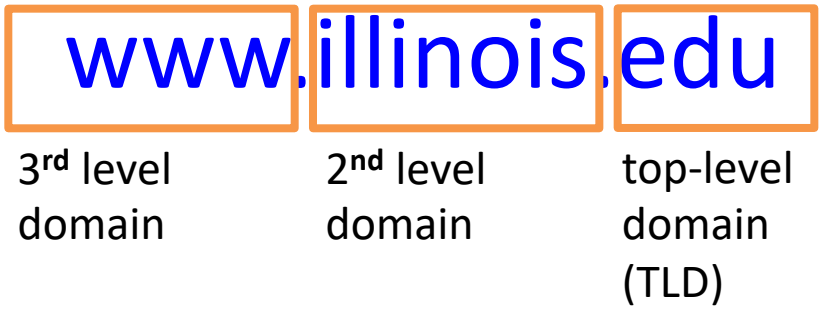
- Application-layer protocols (and people) usually refer to Internet host by *host name*
- Host names organized into hierarchy

www.illinois.edu

top-level
domain (TLD)

Domain Name System

- Application-layer protocols (and people) usually refer to Internet host by *host name*
- Host names organized into hierarchy



DNS Hierarchy

- Each level allocates names to next level
- TLDs allocated by ICANN
  - ccTLD: country-based TLDs, two letters (e.g. `.us`)
  - gTLD: arbitrary names, 3+ letters (e.g. `.com`)
- TLD operated by different registries
- Registrars are agents that register domains for a person or organization in a particular TLD
- Organizations have control over its subdomains
  - e.g. UIUC decides what domains have suffix illinois.edu

Domain Name System

- **Domain Name System (DNS)** is at once:
  - Administrative structure for controlling names
  - Global distributed database of names
  - Protocol for interacting with database
- In this class we will only discuss the security of the DNS protocol
  - There are security problems beyond the protocol
- Still need to understand how protocol is used

Domain Name System

- **DNS Record:** Unit of information in DNS
  - Type: type of data it contains
  - TTL: time to live
- **A record:** IP address for a host name
- **NS record:** Authority name server for a domain
- **MX record:** SMTP (mail) server for domain

DNS Records

- A DNS server has a set of records for which it is the authoritative source

```
$ dig bob.ucsd.edu

;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30439
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; QUESTION SECTION:
;bob.ucsd.edu.                IN   A

;; ANSWER SECTION:
bob.ucsd.edu.      3600 IN   A    132.239.80.176

;; AUTHORITY SECTION:
ucsd.edu.      3600 IN   NS   ns0.ucsd.edu.
ucsd.edu.      3600 IN   NS   ns1.ucsd.edu.
ucsd.edu.      3600 IN   NS   ns2.ucsd.edu.
```

DNS Records

- A DNS server has a set of records for which it is the authoritative source

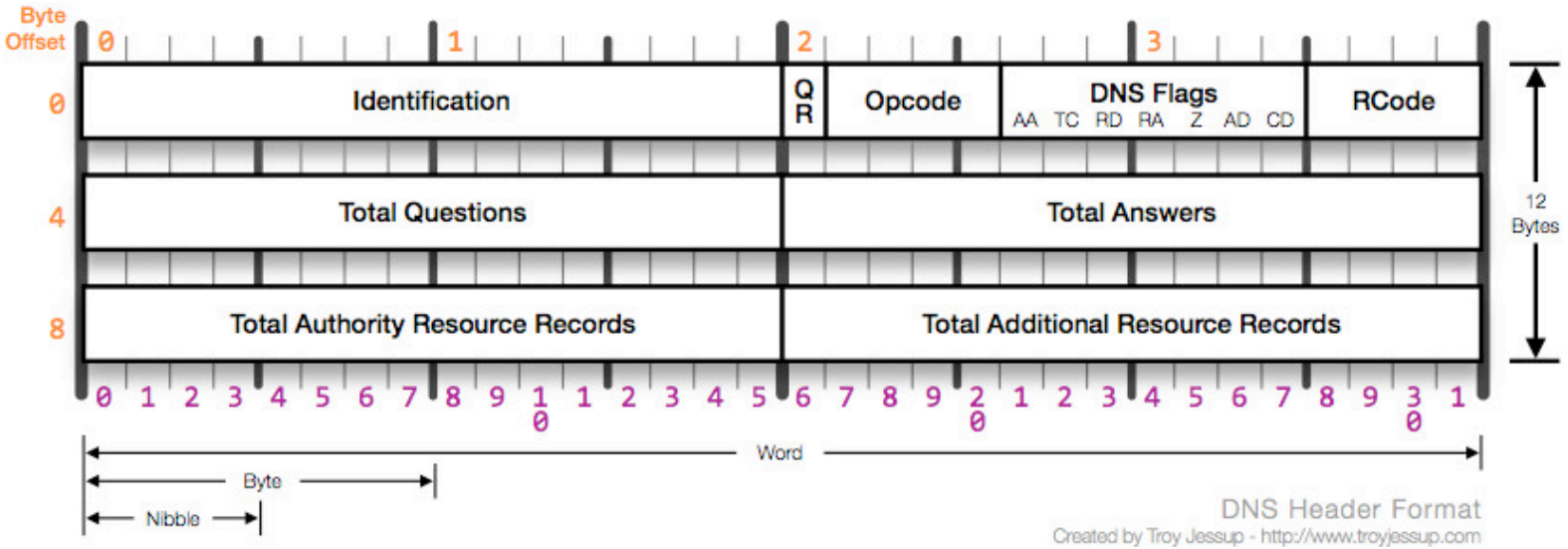
```
$ dig bob.ucsd.edu

;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30439
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; QUESTION SECTION:
;bob.ucsd.edu.                  IN   A

;; ANSWER SECTION:
bob.ucsd.edu.       3600 IN   A    132.239.80.176

;; AUTHORITY SECTION:
ucsd.edu.      3600 IN   NS   ns0.ucsd.edu.
ucsd.edu.      3600 IN   NS   ns1.ucsd.edu.
ucsd.edu.      3600 IN   NS   ns2.ucsd.edu.
```

TTL

type

DNS Protocol

- Uses UDP as transport
- 16-bit query ID to match response to query
- Four sections:
  questions, *answers*, *authority*, *additional records*
- No encryption or authentication of data or header



DNS Records

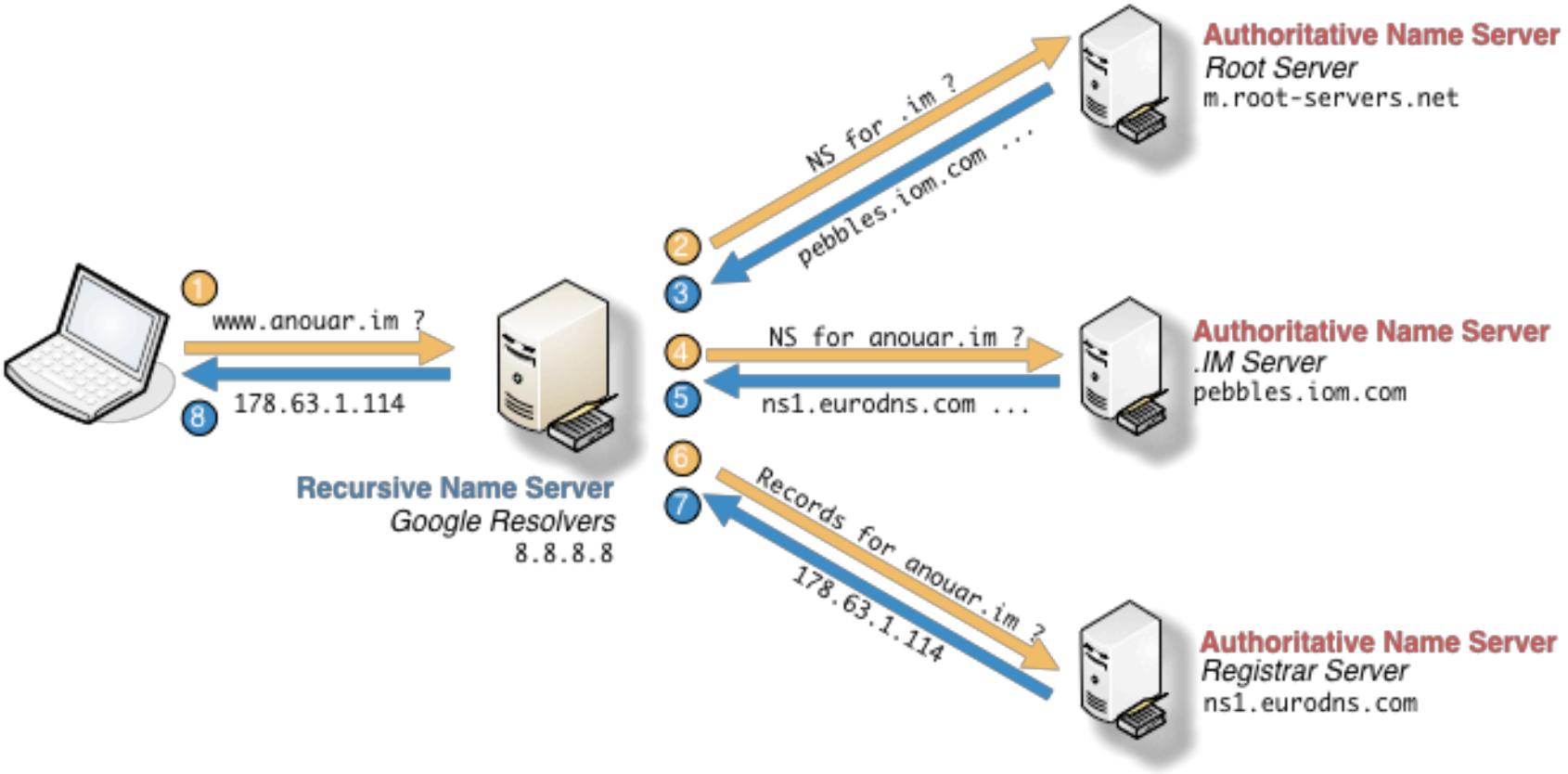
- A DNS server has a set of records for which it is the authoritative source

```
$ dig bob.ucsd.edu

;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30439
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; QUESTION SECTION:
;bob.ucsd.edu.                  IN   A

;; ANSWER SECTION:
bob.ucsd.edu.       3600 IN   A    132.239.80.176

;; AUTHORITY SECTION:
ucsd.edu.      3600 IN   NS   ns0.ucsd.edu.
ucsd.edu.      3600 IN   NS   ns1.ucsd.edu.
ucsd.edu.      3600 IN   NS   ns2.ucsd.edu.
```

TTL

type

DNS Name Resolution



source: http://anouar.adlani.com/2011/12/useful-dig-command-to-troubleshot-your-domains.html

DNS Server Roles

- **Authoritative server:** provides authoritative information for a set of domains
  - Does not handle queries about other domains
- **Recursive resolver:** provides recursive resolution of a domain to return requested record to client
  - Handles queries about all domains
- Same protocol for both types of servers
  - Distinction is in intended purpose only

DNS Security Properties

	Passive	Off-Path	MitM
Availability	—	✗	✗
Confidentiality		—	
Integrity	—	—	
Authenticity	—		

- What properties DNS provide?

DNS Security Properties

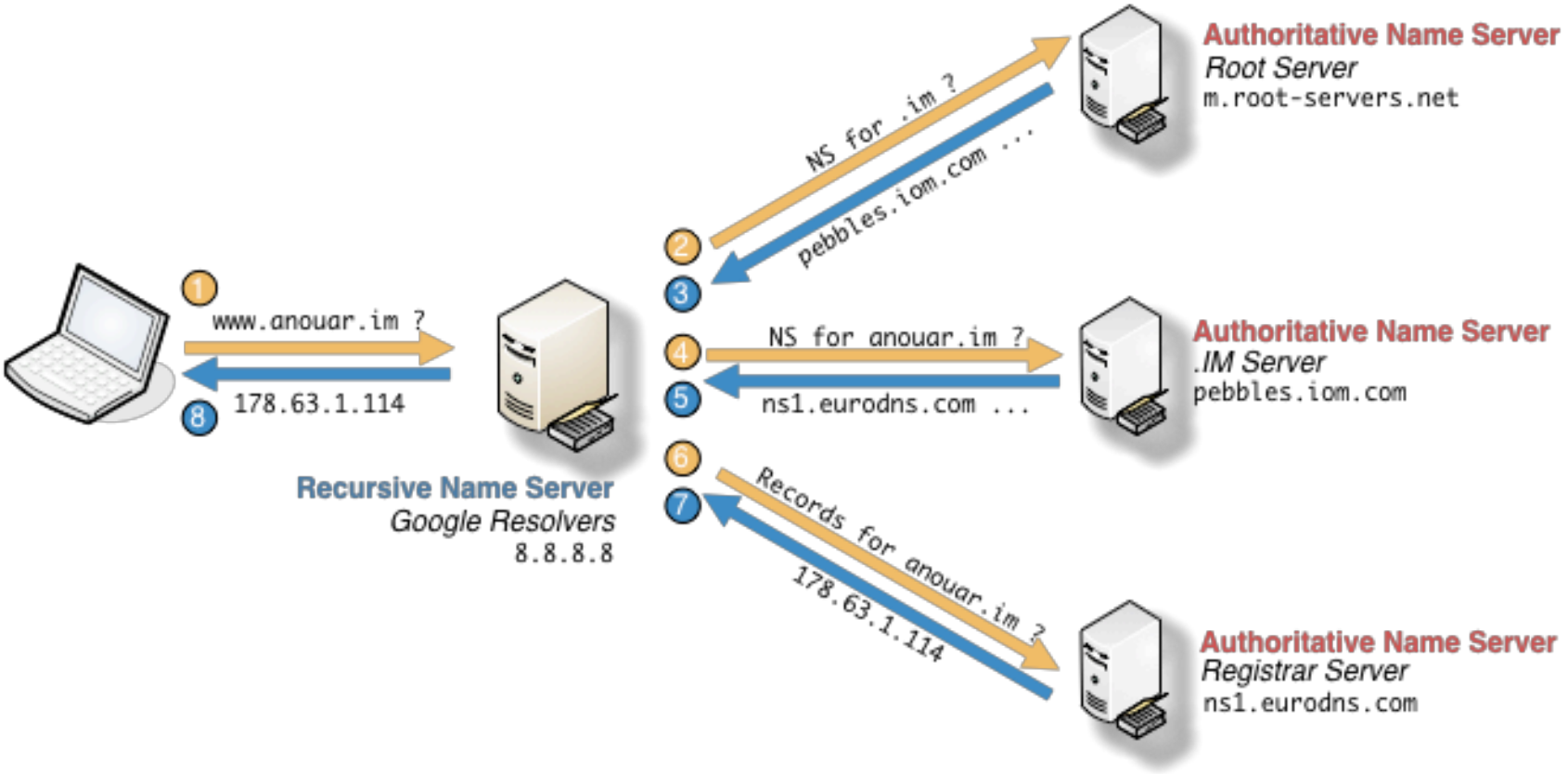
	Passive	Off-Path	MitM
Availability	—	✗	✗
Confidentiality	✗	—	✗
Integrity	—	—	✗
Authenticity	—		✗

- **MitM:** no additional protection
- **Passive:** no additional protection
- What about off-path attacks?

DNS Off-Path Attacks

- Scenario: DNS client issues query to server
- Attacker would like to inject a fake reply

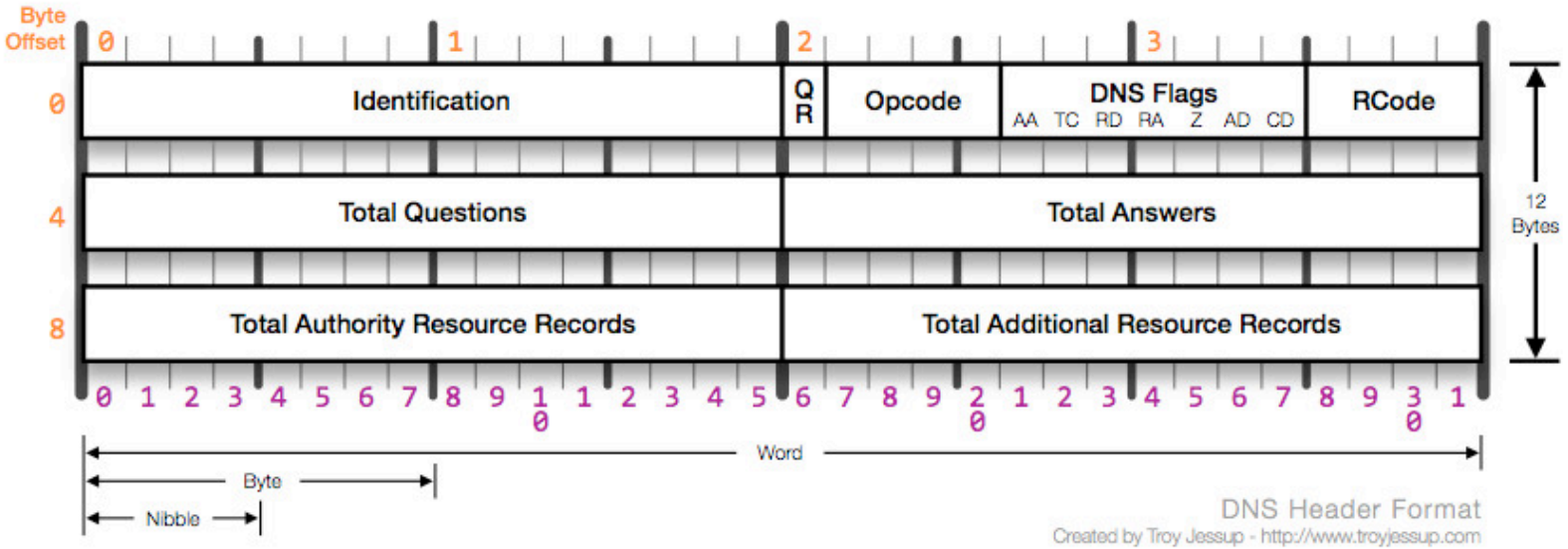
DNS Name Resolution



source: http://anouar.adlani.com/2011/12/useful-dig-command-to-troubleshot-your-domains.html

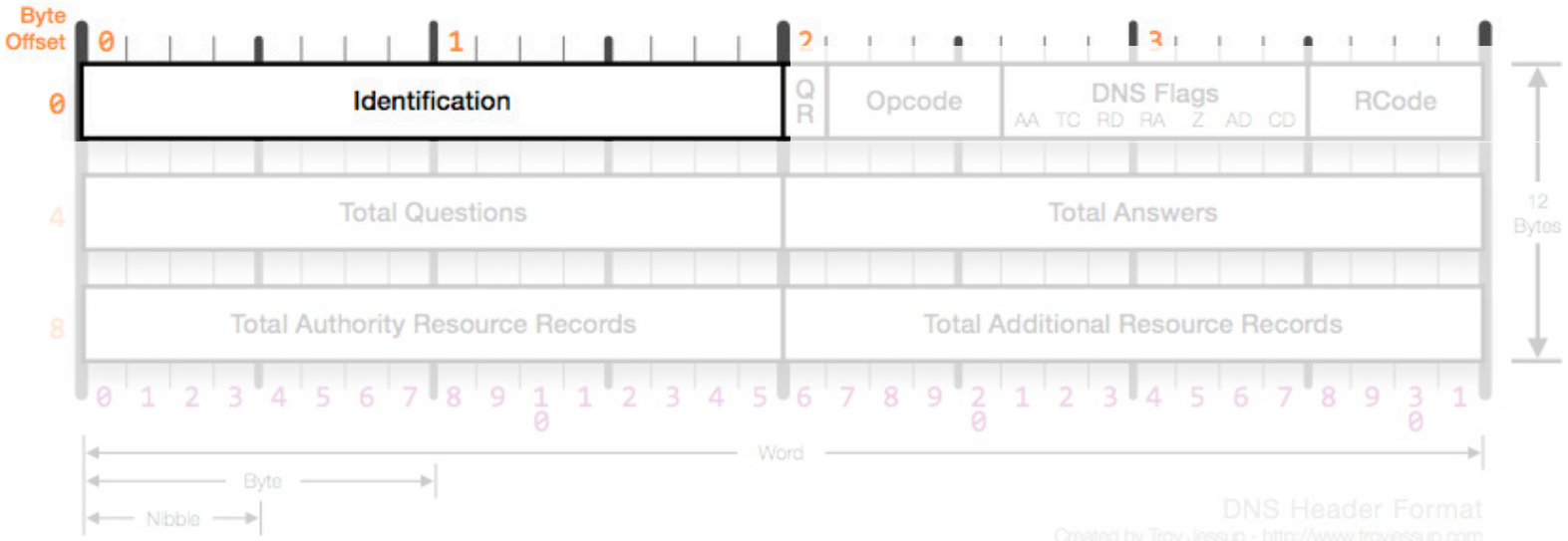
DNS Spoofing Attack

- Scenario: DNS client issues query to server
- Attacker would like to inject a fake reply
  - Attacker does not see query or real response
- How does client authenticate response?



DNS Spoofing Attack

- How does client authenticate response?
  - UDP port numbers must match
    - Destination port usually port 53 by convention
  - 16-bit query ID must match



DNS Spoofing Attack

- Randomized UDP source port: 2^{-32}
- Fixed UDP source port: 2^{-16}
  - Many DNS resolvers (e.g. BIND) re-used the same socket for all queries for performance reasons
    - Source port the same across queries
- **But:** need to time response exactly to arrive after query issued but before real response arrives
- **But:** DNS resolvers cache replies until TTL expires
- *DNS spoofing thought to be mitigated by the above points even for servers that reused query socket*

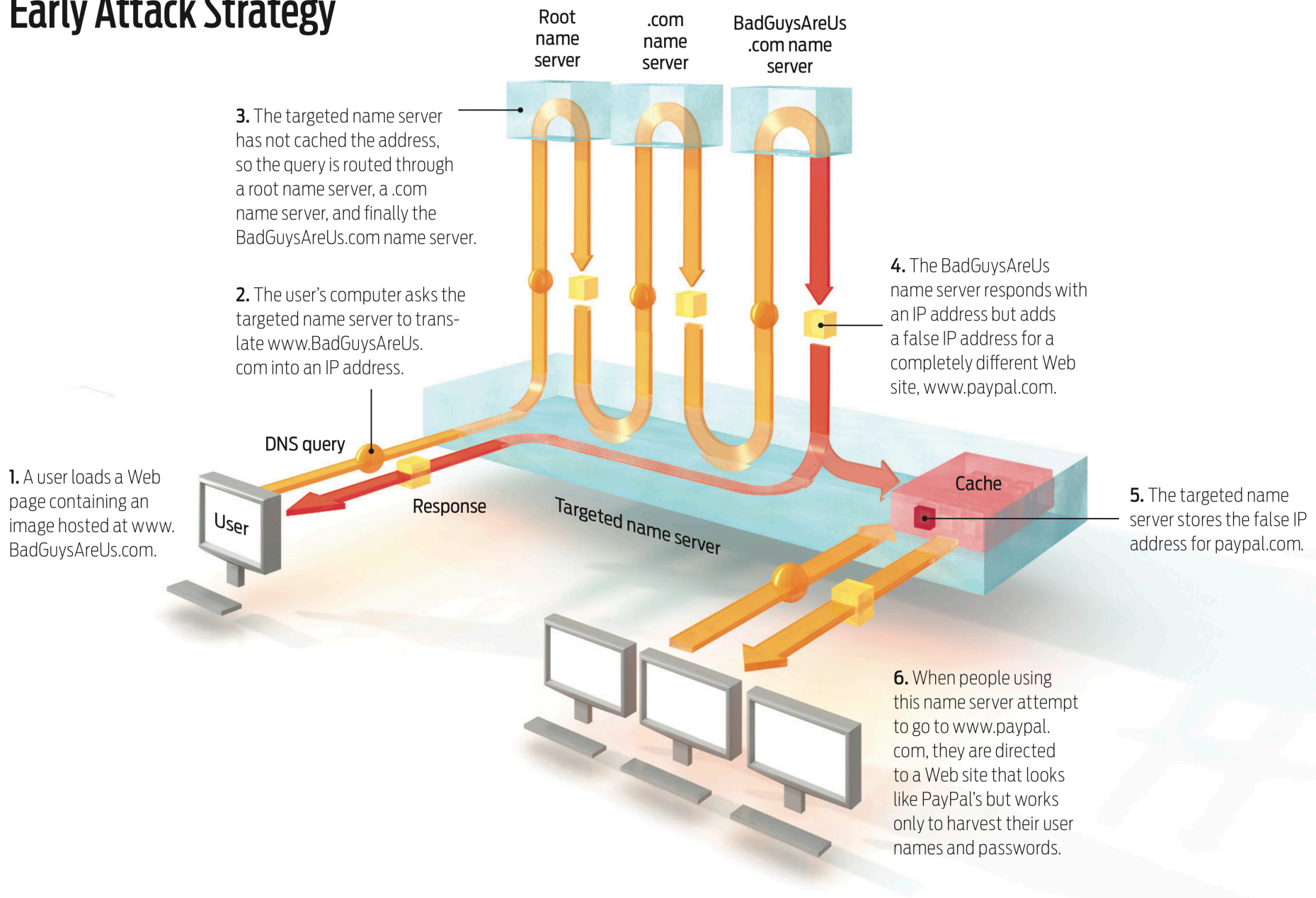
DNS Caching

- Recursive resolvers cache records to avoid repeating recursive resolution process for each query
- Lifetime of record determined by record TTL
  - Could also be evicted from cache due to limited memory
- Injecting spoofed records into a resolver's cache is called *DNS cache poisoning*
  - No protocol-defined way for to refresh cached record

DNS Cache Poisoning

- DNS query results include Additional Records section
  - Provide records for anticipated next resolution step
- Early servers accepted and cached all additional records provided in query response
  - What is wrong with this?

Early Attack Strategy



DNS Cache Poisoning

- DNS query results include Additional Records section
  - Provide records for anticipated next resolution step
- Early servers accepted and cached all additional records provided in query response
  - What is wrong with this?
- Can we just stop using additional section?

DNS Glue Records

- Can we just stop using additional section?
- **Glue records:** non-authoritative records needed to contact next hop in resolution chain
  - Necessary given current design of DNS

```
; <<>> DiG 9.6-ESV-R4-P3 <<>> @192.5.6.30 ucsd.edu
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12781
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 3, ADDITIONAL: 4
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;ucsd.edu.                      IN      A

;; AUTHORITY SECTION:
ucsd.edu.               172800  IN      NS      ns1.ucsd.edu.
ucsd.edu.               172800  IN      NS      ns2.ucsd.edu.
ucsd.edu.               172800  IN      NS      ns0.ucsd.edu.

;; ADDITIONAL SECTION:
ns1.ucsd.edu.           172800  IN      A       128.54.16.2
ns2.ucsd.edu.           172800  IN      A       132.239.1.52
ns0.ucsd.edu.           172800  IN      A       132.239.1.51
ns0.ucsd.edu.           172800  IN      AAAA    2607:f720:100:100::231
```

edu authority

Names of ucsd.edu authoritative servers

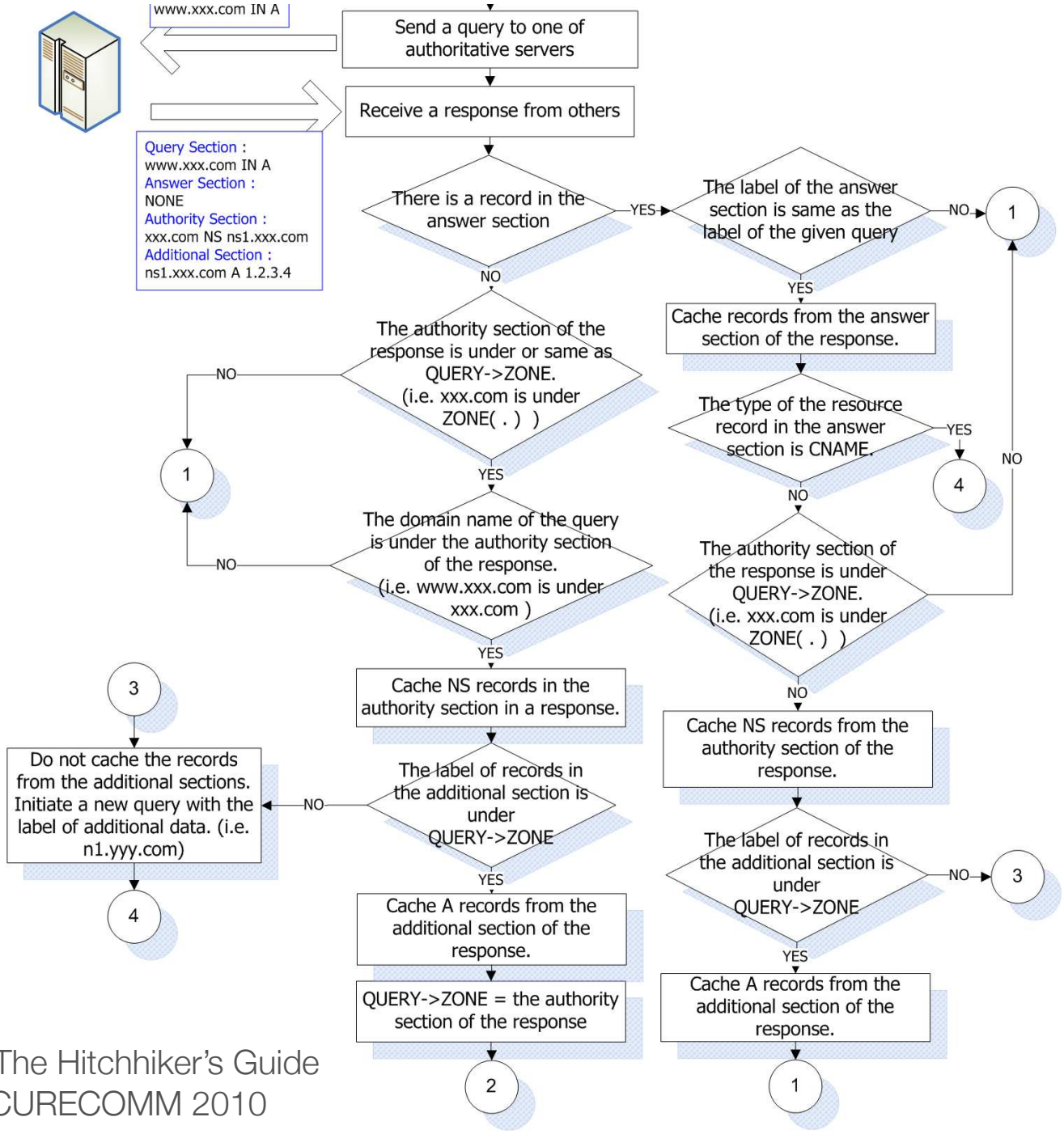
Glue records for authoritative servers

What should be the policy for accepting additional records?

Bailiwick Rules

- **Bailiwick rule:** defines what response records a recursive resolver will accept
- **Bailiwick:** set of domains about which a server is has direct or indirect authority to speak
  - Bailiwick determined by the initiator of query
- Answer should be relevant *(in response to request)*
- Answer should be in bailiwick

Bailiwick Checking Rule from BIND



source: Son and Shmatikov, “The Hitchhiker’s Guide to DNS Cache Poisoning” SECURECOMM 2010

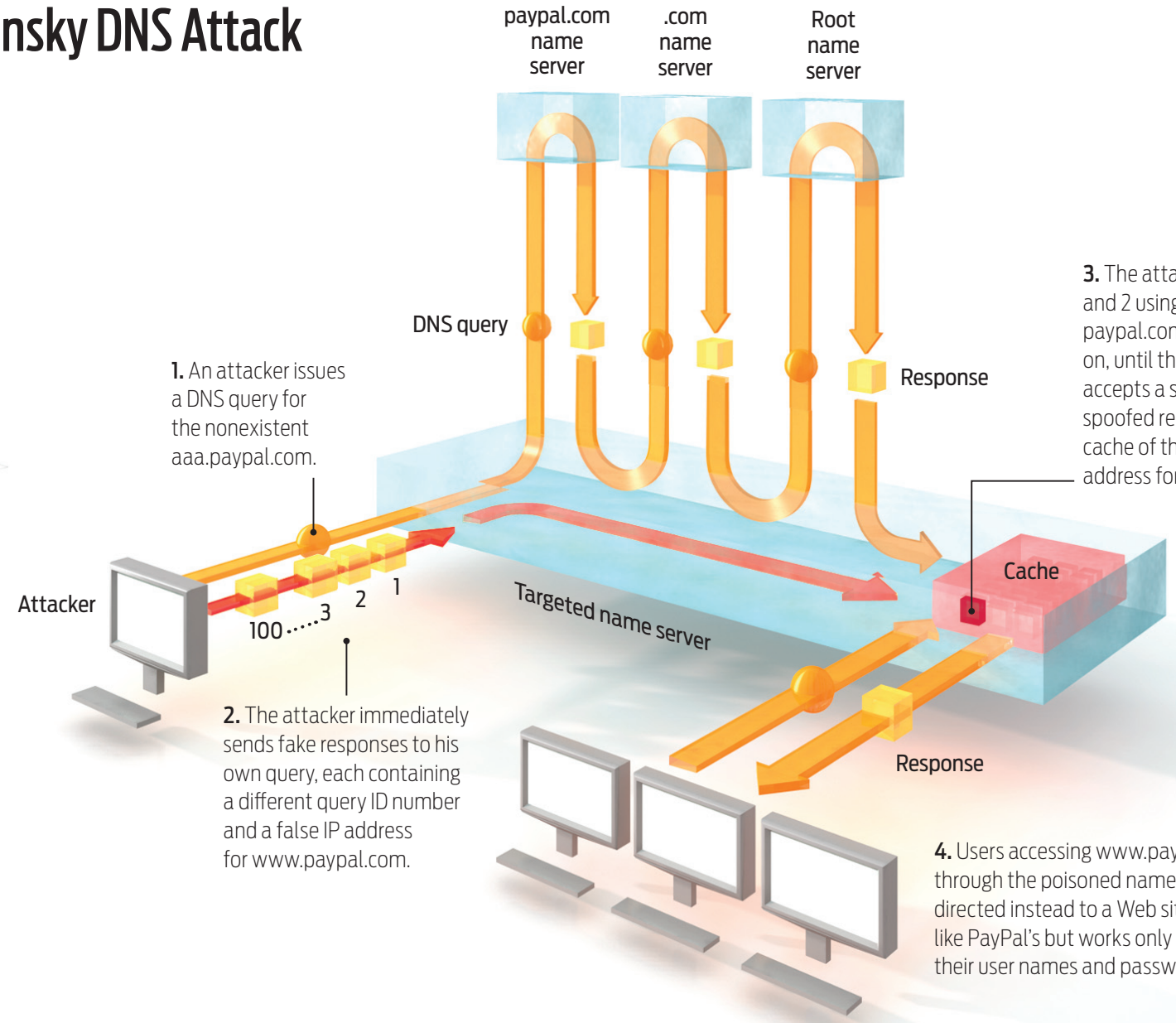
BIND Bailiwick Rule (roughly)

- Authorities must be for queried domain
  - `dns1.illinois.edu` accepted as authority for illinois.edu only when initiating query was for subdomain of `illinois.edu`
- Additional records must be *in bailiwick* for query
  - A record for `dns1.illinois.edu` accepted because edu server has indirect authority over `dns1.illinois.edu`

Kaminsky's Attack

- Naive attacks on DNS are throttled by TTL
- Each result of query cached for TTL duration
  - Future attack queries answered from cache, so can't poison until cache expires
- **Attack:** inject forged *additional records* for queries to non-existent subdomains
- This bypasses TTL throttling

Kaminsky DNS Attack



BRYAN CHRISTIE DESIGN

Kaminsky's Attack

- If attacker can send *N* spoofed response packets for each query, chance of guessing
  - Randomized source port: $N \times 2^{-32}$
  - Fixed source port: $N \times 2^{-16}$
- Can immediately repeat attack again
  - Not throttled by TTL!
- Need to make the resolver whose cache we want to poison issue a query for subdomain in target domain
  - So this is not *strictly* an off-path attack

DNS Security Properties

	Passive	Off-Path	MitM
Availability	—	✗	✗
Confidentiality	✗	—	✗
Integrity	—	—	✗
Authenticity	—		✗

- Spoofing DNS with proper randomization: 2^{-32}
- Bailiwick rule prevents direct cache poisoning