Domain Name Systems

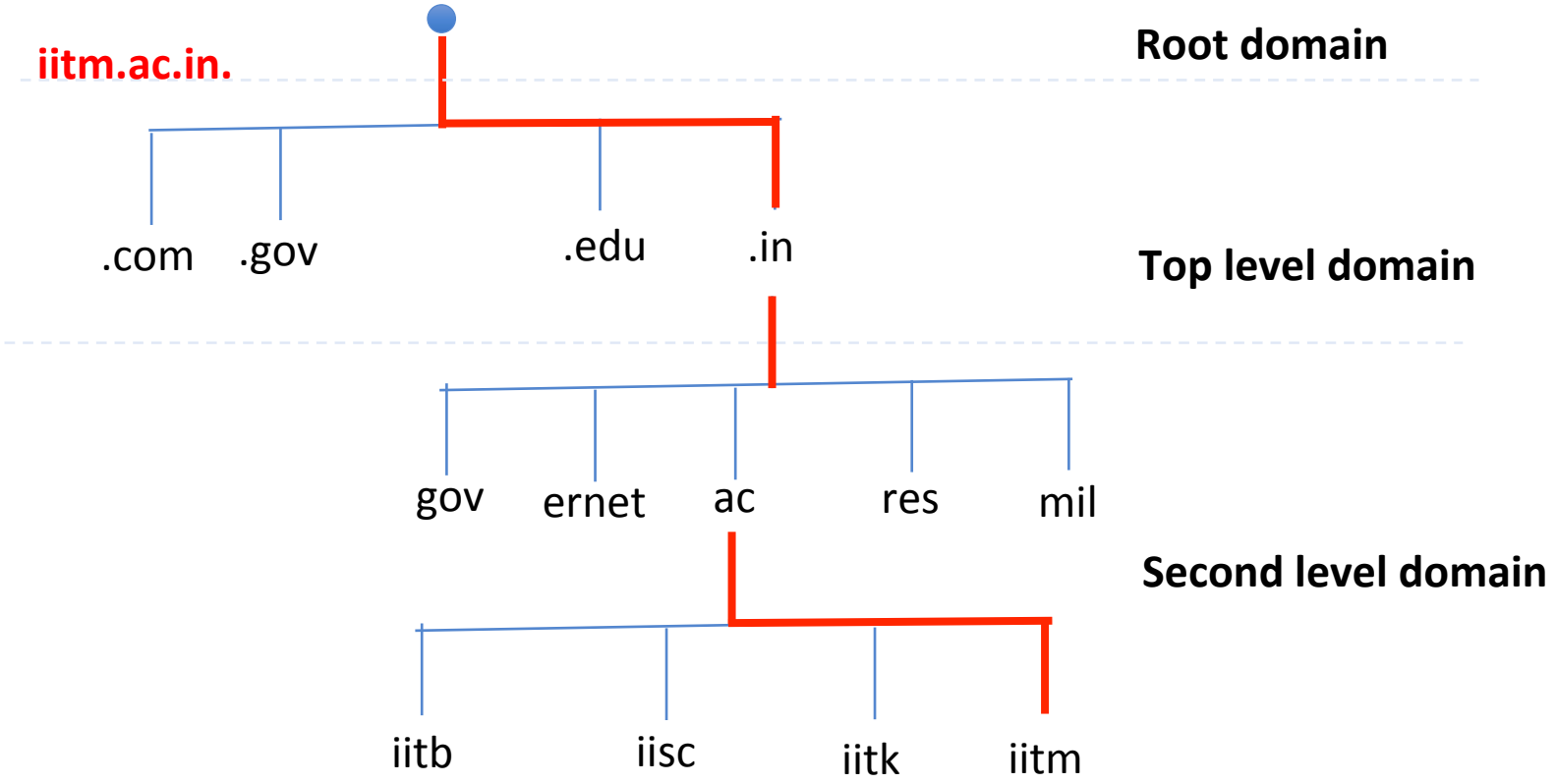
Chester Rebeiro

IIT Madras

Some of the slides borrowed from the book ‘Computer Security: A Hands on Approach’ by Wenliang Du

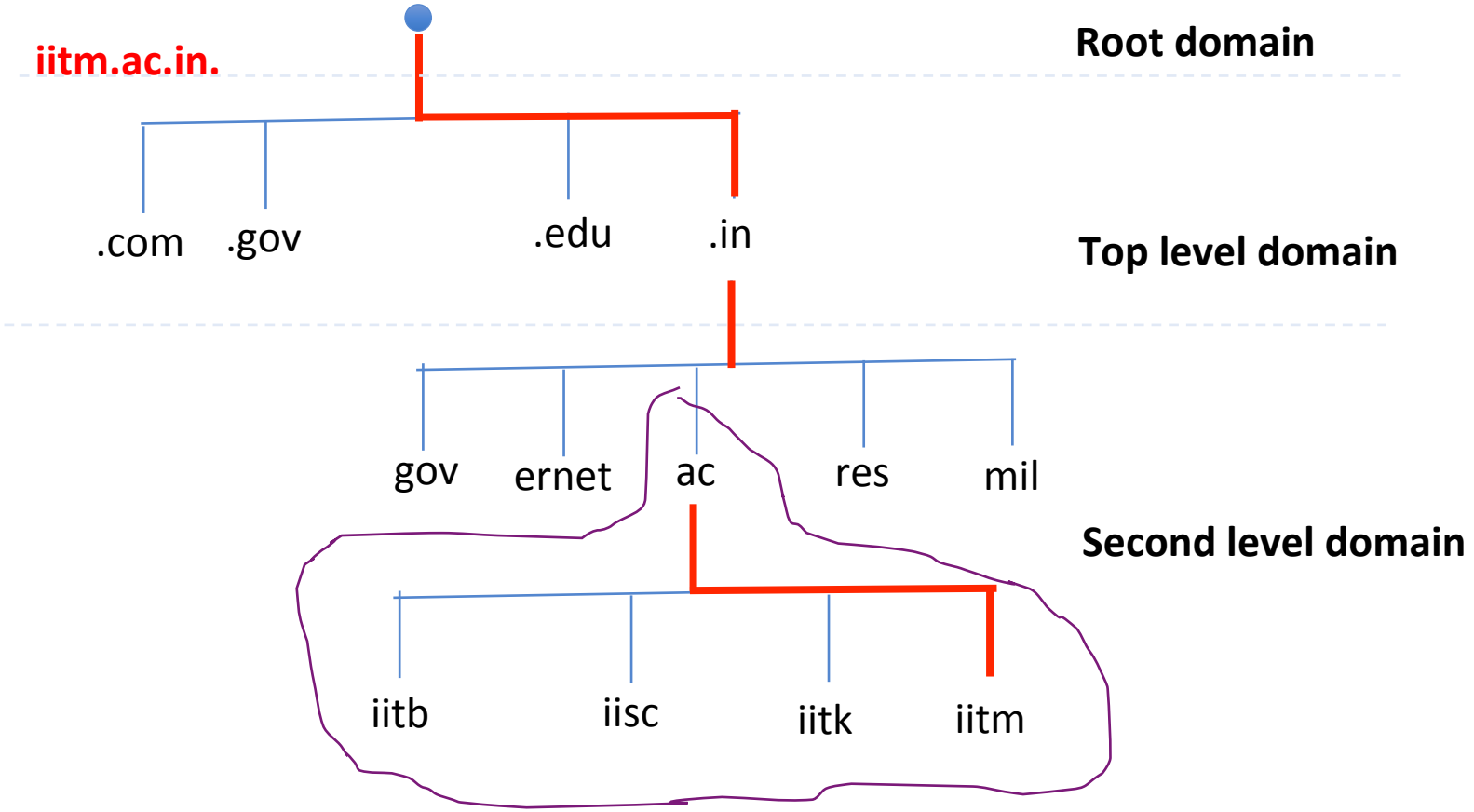
DNS Hierarchy

Lookup records for mapping from domain names to IP addresses



DNS Hierarchy

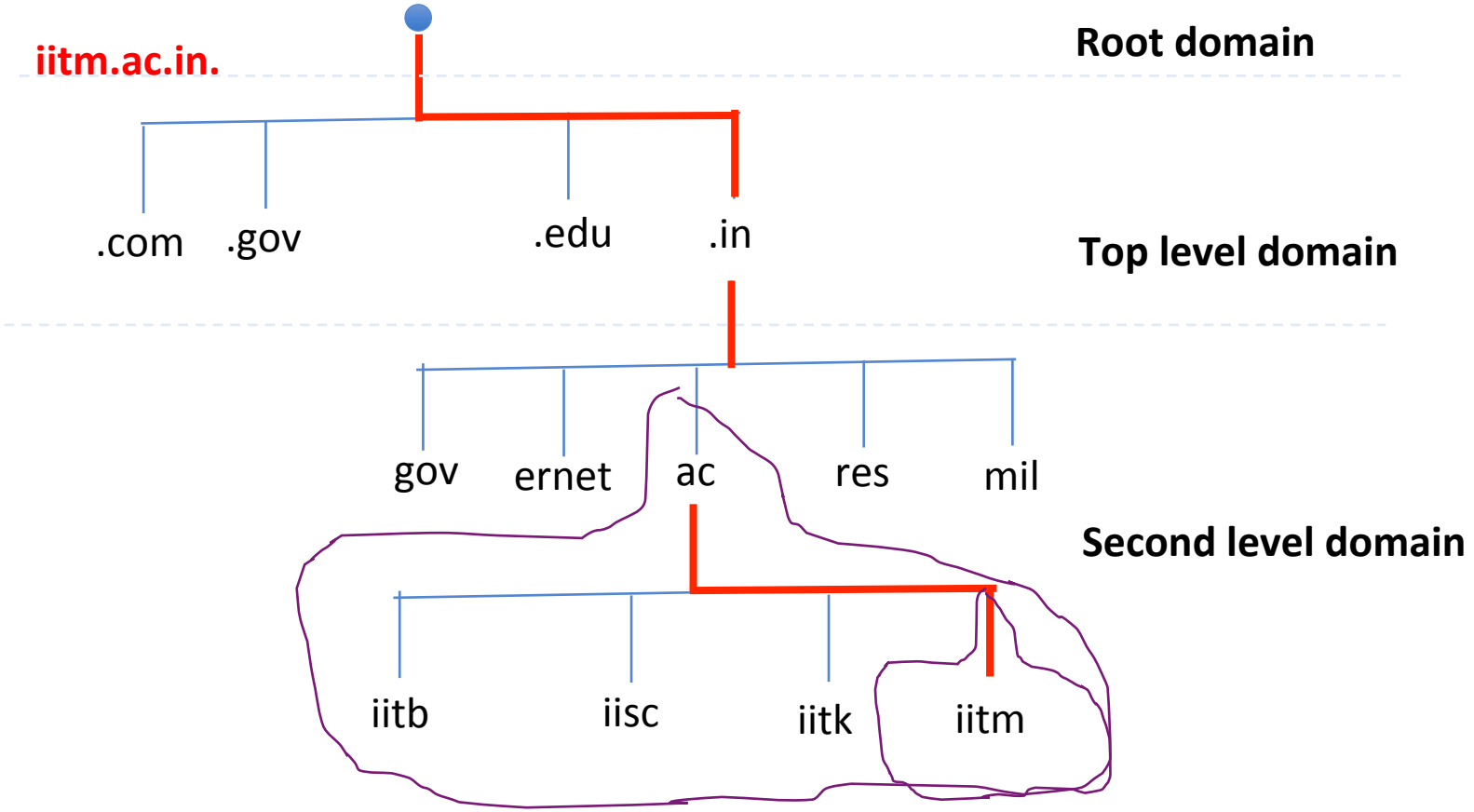
Lookup records for mapping from domain names to IP addresses



Domain:
Is a subtree, sharing its domain name with the name of the top most node in the subtree

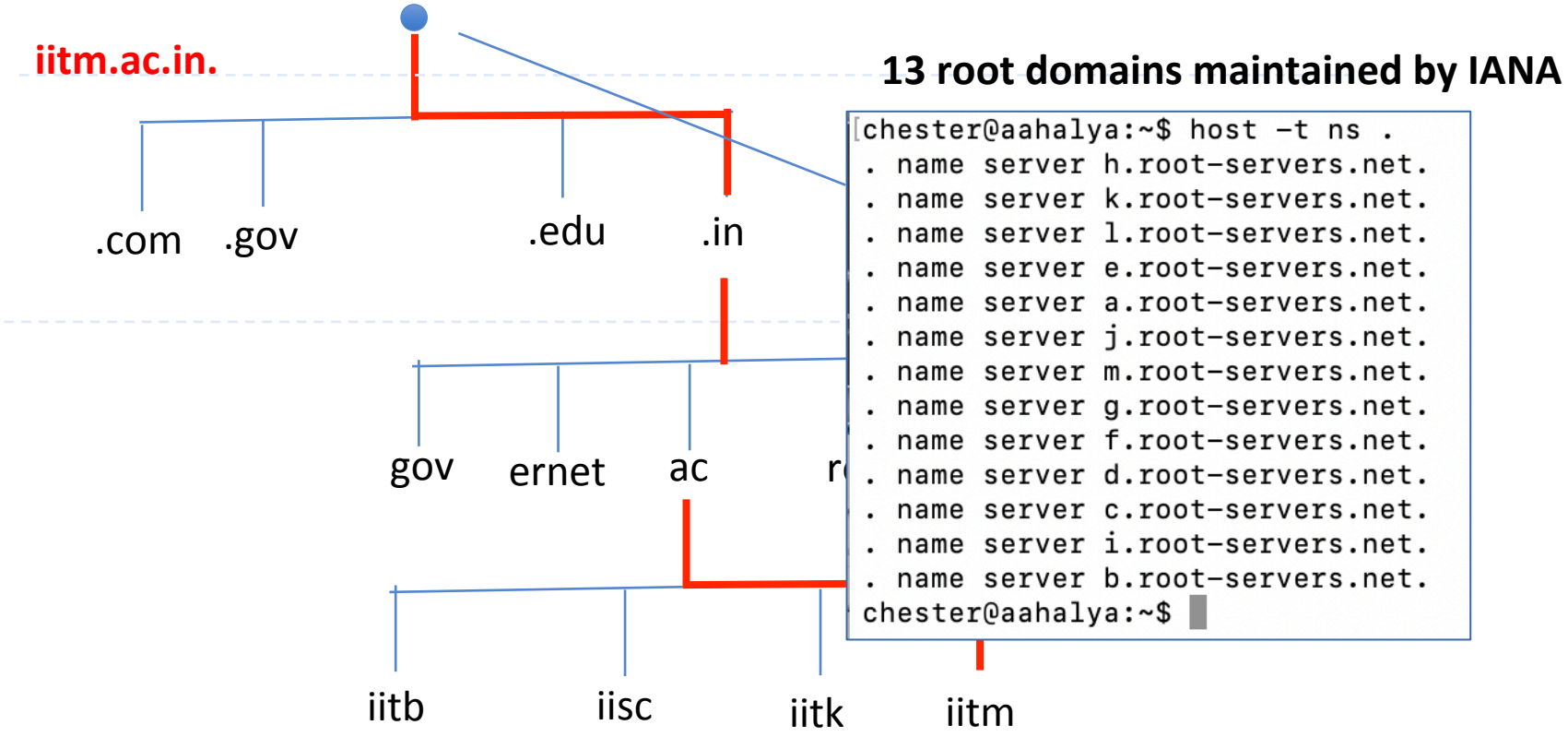
DNS Hierarchy

Lookup records for mapping from domain names to IP addresses



SubDomains:
Is a domain that branches off another.

Root Domain



13 root domains maintained by IANA

```
chester@aahalya:~$ host -t ns .
. name server h.root-servers.net.
. name server k.root-servers.net.
. name server l.root-servers.net.
. name server e.root-servers.net.
. name server a.root-servers.net.
. name server j.root-servers.net.
. name server m.root-servers.net.
. name server g.root-servers.net.
. name server f.root-servers.net.
. name server d.root-servers.net.
. name server c.root-servers.net.
. name server i.root-servers.net.
. name server b.root-servers.net.
chester@aahalya:~$
```

Root Domain

List of Root Servers

HOSTNAME	IP ADDRESSES	MANAGER
a.root-servers.net	198.41.0.4, 2001:503:ba3e::2:30	VeriSign, Inc.
b.root-servers.net	199.9.14.201, 2001:500:200::b	University of Southern California (ISI)
c.root-servers.net	192.33.4.12, 2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13, 2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10, 2001:500:a8::e	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241, 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4, 2001:500:12::d0d	US Department of Defense (NIC)
h.root-servers.net	198.97.190.53, 2001:500:1::53	US Army (Research Lab)
i.root-servers.net	192.36.148.17, 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30, 2001:503:c27::2:30	VeriSign, Inc.
k.root-servers.net	193.0.14.129, 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42, 2001:500:9f::42	ICANN
m.root-servers.net	202.12.27.33, 2001:dc3::35	WIDE Project

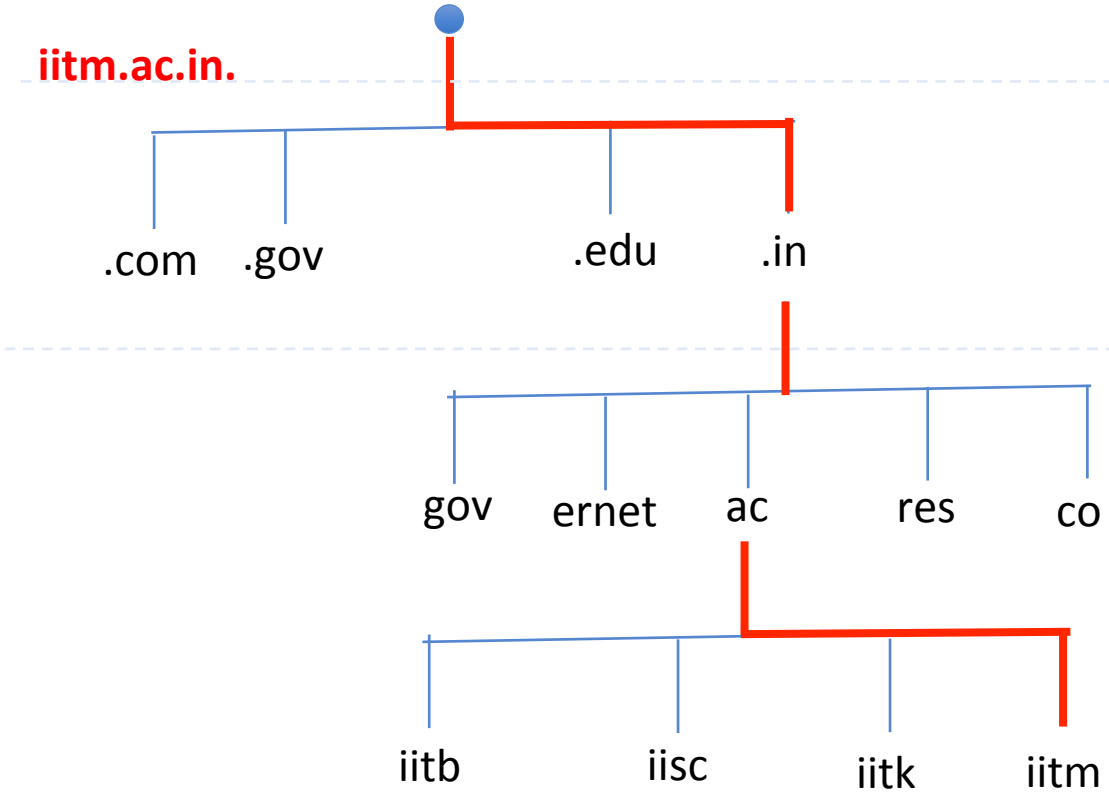
10 in USA
1 in Netherlands
1 in Sweden
1 in Japan

Why only 13 root servers?

567 mirrored root servers
(9 mirrors in India – 2015)
(3 J root servers; 2 L root servers;
1 I, 1 K, 1 F, and D root server)

https://internetdemocracy.in/wp-content/uploads/2016/03/Dr.-Anja-Kovacs-and-Rajat-Rai-Handa-India-at-the-Internets-Root.pdf

Top Level Domains



Top level domain

1547 as on July 2017

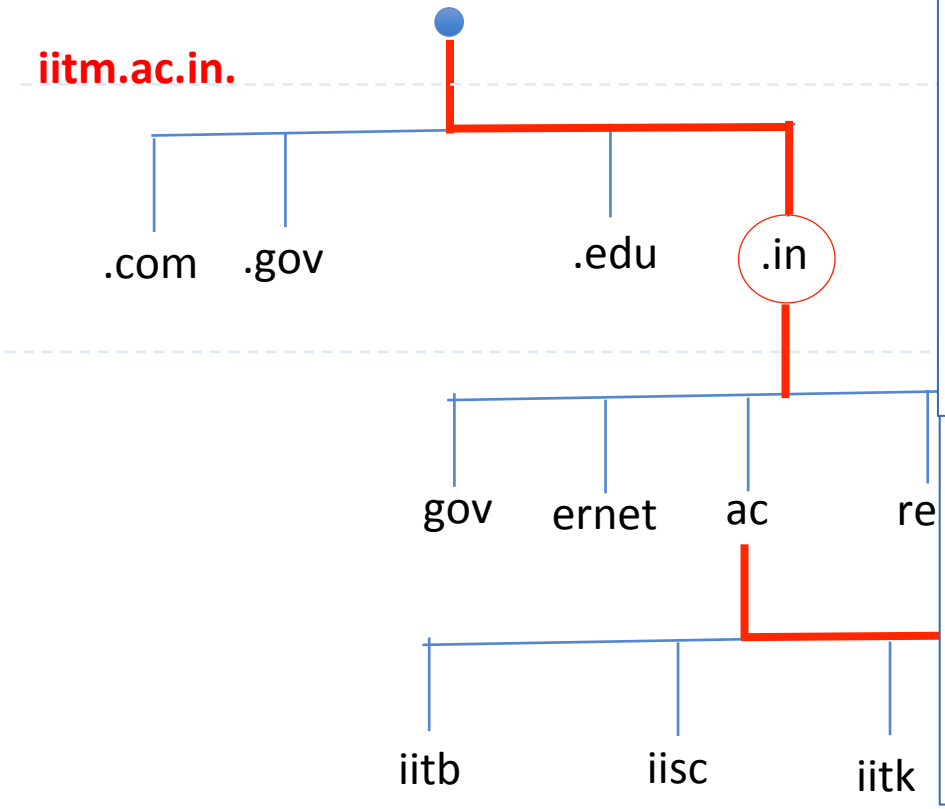
Each TLD is managed by designated entities called registries.
(for example: .com, .net is managed by Verisign;
.in is managed by National Internet Exchange of India)

https://en.wikipedia.org/wiki/INRegistry

Top Level Domains

Top level domain

1547 as on July 2017

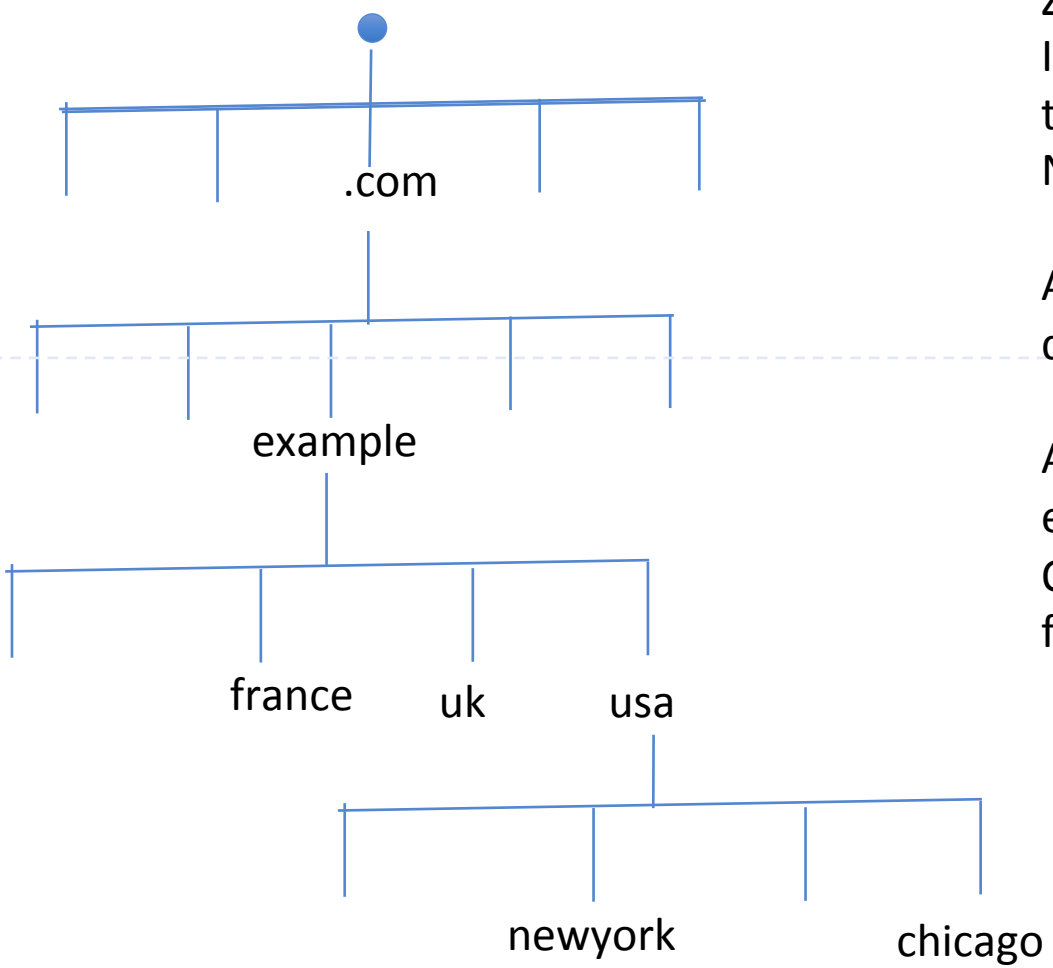


- .in (available to anyone; used by companies, individuals, and organizations in India)
- .co.in (originally for banks, registered companies, and trademarks)
- .firm.in (originally for shops, partnerships, liaison offices, sole proprietorships)
- .net.in (originally for Internet service providers)
- .org.in (originally for non-profit organisations)
- .gen.in (originally for general/miscellaneous use)
- .ind.in (originally for individuals,companies)

- .ac.in (Academic institutions)
- .edu.in (Educational institutions)
- .res.in (Indian research institutes)
- .ernet.in (Older, for both educational and research institutes)
- .gov.in (Indian government)
- .mil.in (Indian military)

https://en.wikipedia.org/wiki/INRegistry

DNS Zone



Zone:
Is a domain (or subdomain) that branches is served by a Name server.

A zone may be an entire domain with all its child domains, or a portion of a domain.

A zone can be the entire subtree starting at example.com
Or the company may decide to have several sub zones, for example one at usa.example.com

Authoritative Name Servers

```
chester@aahalya:~$ nslookup
> set querytype=soa
> iitm.ac.in
Server:         10.6.0.11
Address:        10.6.0.11#53

Non-authoritative answer:
iitm.ac.in
        origin = dns1.iitm.ac.in
        mail addr = root.dns1.iitm.ac.in
        serial = 2019020801
        refresh = 10800
        retry = 3600
        expire = 1814400
        minimum = 86400

Authoritative answers can be found from:
iitm.ac.in      nameserver = dns2.iitm.ac.in.
iitm.ac.in      nameserver = dns1.iitm.ac.in.
dns1.iitm.ac.in internet address = 10.24.4.10
dns2.iitm.ac.in internet address = 10.24.4.11
```

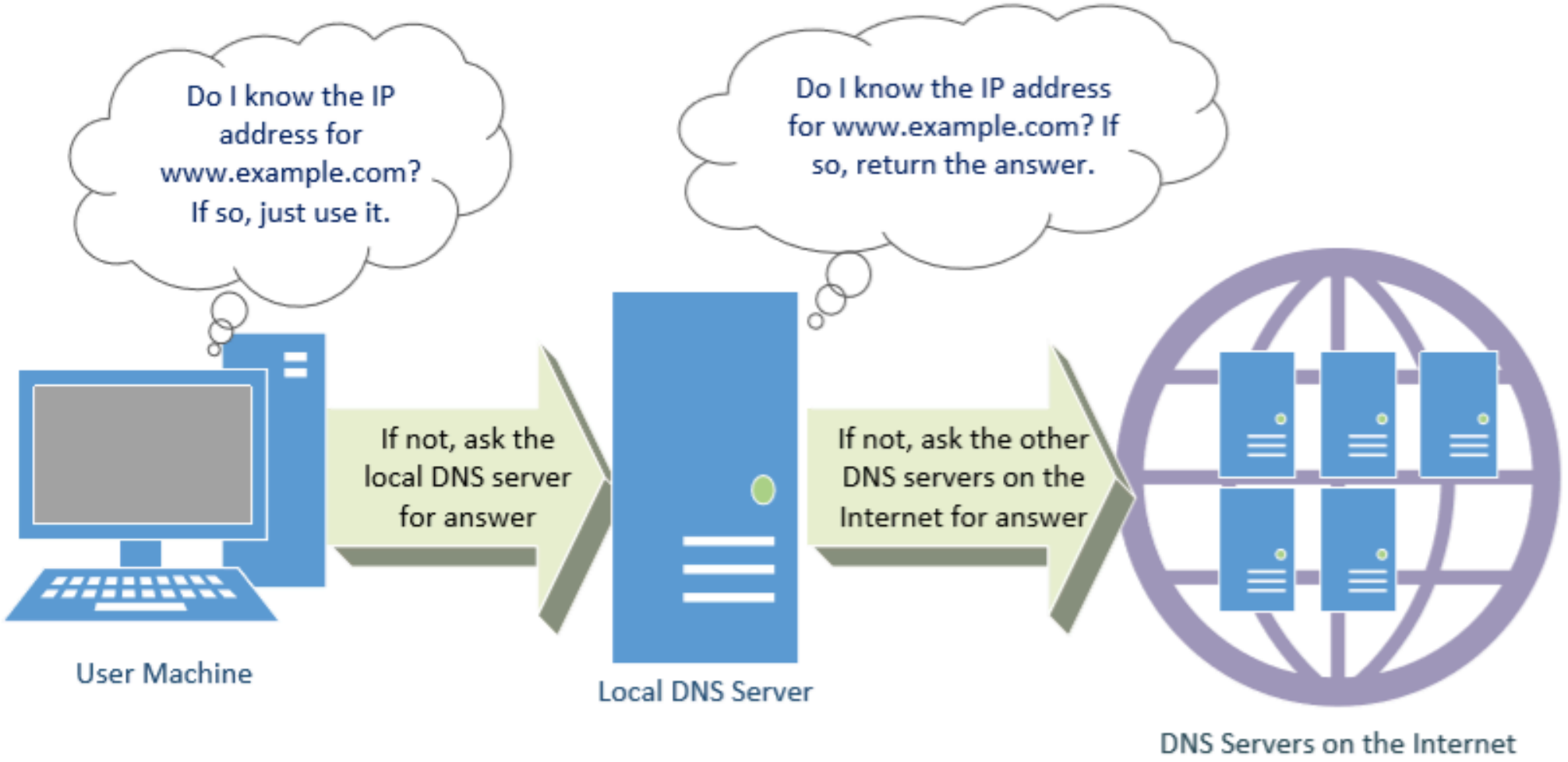
Start of authority

Each DNS Zone has at least one authoritative name server that publishes information about that zone.

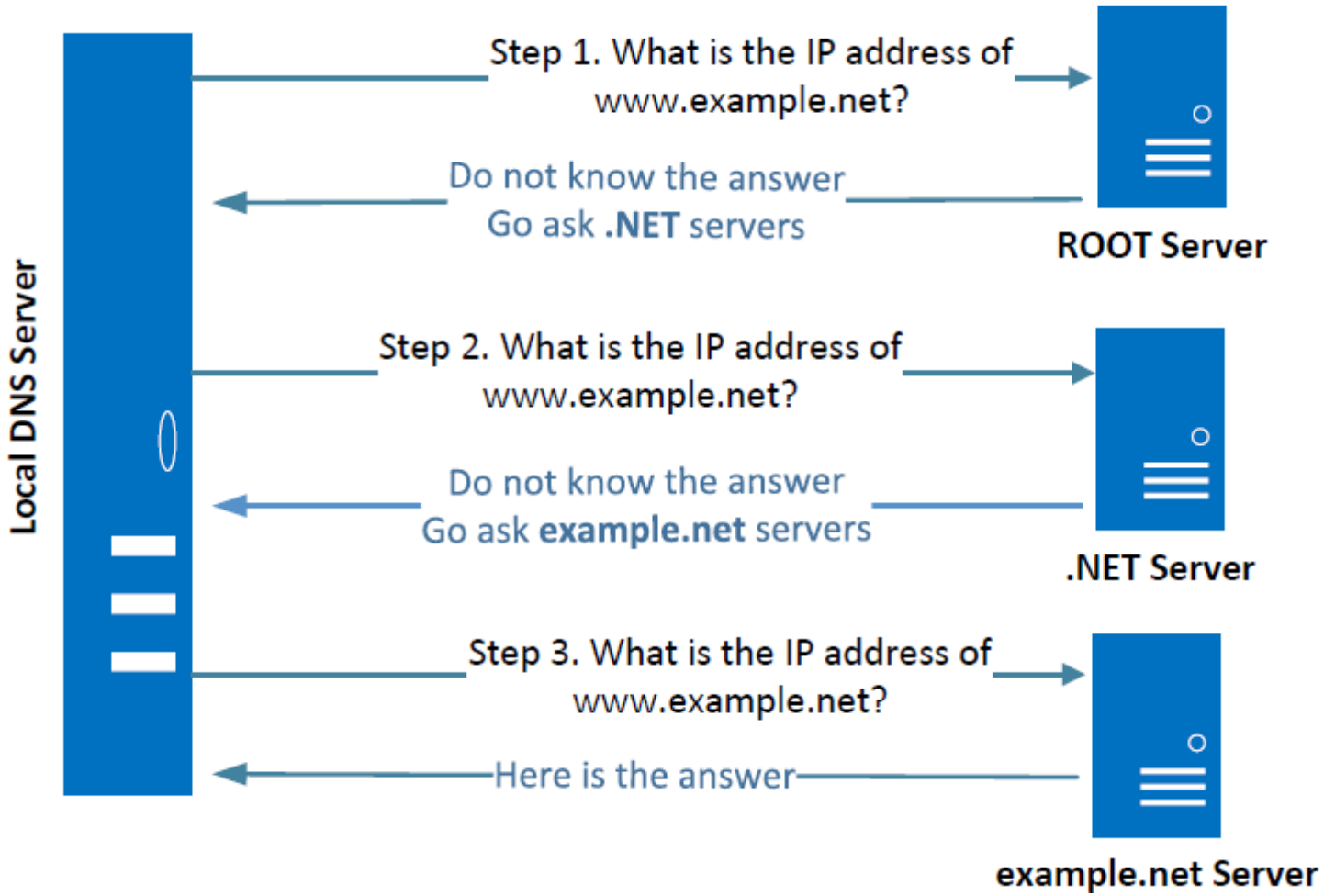
They are called `authoritative' because they provide original and answers to DNS queries as opposed to obtaining answers from other DNS servers.

2 authorities. 1 primary and the other secondary

DNS Query Process



Local DNS Server and Iterative Query



The iterative process starts from the ROOT Server. If it doesn't know the IP address, it sends back the IP address of the nameservers of the next level server (.NET server) and then the last level server (example.net) which provides the answer.

DNS Query Process

http://www.iitm.ac.in → Entered in web-browser

1

Local System:
Lookup /etc/hosts file.
Can the /etc/hosts file resolve (have the IP address) for www.iitm.ac.in?

```
chester@aahalya:~$ cat /etc/hosts
127.0.0.1       localhost
10.6.15.91      aahalya.cse.iitm.ac.in  aahalya
10.6.15.92      aampal.cse.iitm.ac.in   aampal
10.6.15.93      aarunya.cse.iitm.ac.in  aarunya
10.6.15.94      aarika.cse.iitm.ac.in   aarika
10.6.15.95      baarika.cse.iitm.ac.in  baarika
10.6.15.96      basavi.cse.iitm.ac.in   basavi
10.6.15.97      cabitha.cse.iitm.ac.in  cabitha
chester@aahalya:~$
```

DNS Query Process

http://www.iitm.ac.in → Entered in web-browser

2

Local DNS Server:
Lookup the local DNS server (server present in the LAN).

How to identify the IP address of the Local DNS server? (/etc/resolv.conf)
This needs to be configured or, can be found, if the system is configured for DHCP, then this file is automatically modified.

```
chester@aahalya:~$ cat /etc/resolv.conf
search cse.iitm.ac.in
nameserver 10.6.0.11
nameserver 10.6.0.12
```

If the local DNS server can resolve the address; then we are done. Else, the resolver would be activated.
The resolver would need to query another DNS server, higher up in the hierarchy.

DNS Query Process

http://www.iitm.ac.in.

3

Resolver in Local DNS will query the Root Name Server

→(from resolver) What is the IP address of www.iitm.ac.in

← (from root server)I don't know the answer, you can ask any of these authorities

.

Directly send the query to this server.

```
chester@aahalya:~$ dig @a.root-servers.net iitm.ac.in

;; QUESTION SECTION:
;iitm.ac.in.                    IN      A

;; AUTHORITY SECTION:
in.                     172800  IN      NS      a0.in.afilias-nst.info.
in.                     172800  IN      NS      a1.in.afilias-nst.in.
in.                     172800  IN      NS      a2.in.afilias-nst.info.
in.                     172800  IN      NS      b0.in.afilias-nst.org.
in.                     172800  IN      NS      b1.in.afilias-nst.in.
in.                     172800  IN      NS      b2.in.afilias-nst.org.
in.                     172800  IN      NS      c0.in.afilias-nst.info.

;; ADDITIONAL SECTION:
a0.in.afilias-nst.info. 172800  IN      A       199.7.87.1
a1.in.afilias-nst.in.   172800  IN      A       115.249.164.142
a2.in.afilias-nst.info. 172800  IN      A       199.249.117.1
b0.in.afilias-nst.org.  172800  IN      A       199.253.56.1
b1.in.afilias-nst.in.   172800  IN      A       180.179.215.70
b2.in.afilias-nst.org.  172800  IN      A       199.249.125.1
c0.in.afilias-nst.info. 172800  IN      A       199.253.57.1
a0.in.afilias-nst.info. 172800  IN      AAAA    2001:500:29::1
a1.in.afilias-nst.in.   172800  IN      AAAA    2001:4528:fff:13::142
a2.in.afilias-nst.info. 172800  IN      AAAA    2001:500:45::1
b0.in.afilias-nst.org.  172800  IN      AAAA    2001:500:50::1
b1.in.afilias-nst.in.   172800  IN      AAAA    2401:8800:411:8::70
b2.in.afilias-nst.org.  172800  IN      AAAA    2001:500:4d::1
c0.in.afilias-nst.info. 172800  IN      AAAA    2001:500:51::1
```

DNS Query Process

http://www.iitm.ac.in.

4

Resolver in Local DNS will query the TLD

→(from resolver) What is the IP address of www.iitm.ac.in

← (return)I don't know the answer, you can ask any of these authorities

.

```
chester@aahalya:~$ dig @a0.in.afilias-nst.info iitm.ac.in

;; QUESTION SECTION:
;iitm.ac.in.                    IN      A

;; AUTHORITY SECTION:
iitm.ac.in.             86400   IN      NS      dns3.iitm.ac.in.
iitm.ac.in.             86400   IN      NS      dns2.iitm.ac.in.
iitm.ac.in.             86400   IN      NS      dns1.iitm.ac.in.

;; ADDITIONAL SECTION:
dns1.iitm.ac.in.        86400   IN      A       203.199.213.2
dns2.iitm.ac.in.        86400   IN      A       14.130.160.3
dns3.iitm.ac.in.        86400   IN      A       14.139.160.2
```

DNS Query Process

http://www.iitm.ac.in.

5

Resolver in Local DNS will query the next level NS

→What is the IP address of www.iitm.ac.in

← The SOA is dns1.iitm.ac.in

.

```
chester@aahalya:~$ dig @dns1.iitm.ac.in iitm.ac.in

; <<>> DiG 9.7.3 <<>> @dns1.iitm.ac.in iitm.ac.in
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15832
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;iitm.ac.in.                    IN      A

;; AUTHORITY SECTION:
iitm.ac.in.             86400   IN      SOA     dns1.iitm.ac.in. root.dns
ac.in. 2019020801 10800 3600 1814400 86400
```

DNS Cache

- The local DNS server will cache all responses from other DNS servers (to reduce queries)

- TTL available with all responses, which determines when the entry would be removed from cache

```
chester@aahalya:~$ dig    www.iitm.ac.in
;; AUTHORITY SECTION:
iitm.ac.in.             41303   IN      NS      dns1.iitm.ac.in.
iitm.ac.in.             41303   IN      NS      dns2.iitm.ac.in.

chester@aahalya:~$ dig    www.iitm.ac.in
;; AUTHORITY SECTION:
iitm.ac.in.             41276   IN      NS      dns2.iitm.ac.in.
iitm.ac.in.             41276   IN      NS      dns1.iitm.ac.in.
```

TTL

DNS Cache

- Due to caching
  - Most resolver queries do not need a query to the root server
  - 2% of all queries to the root-servers are legitimate
    - 75% were due to incorrect or non-existent caching
    - 12.5% to unknown TLDs
    - 7% were for lookups to IP addresses, as if, they were domain names

https://en.wikipedia.org/wiki/Root_name_server

Set Up DNS Zones on Local DNS Server

- Utility in Linux: bind9
- Create zones: Create two zone entries in the DNS server by adding them to `/etc/bind/named.conf`.

```
zone "example.net" {
        type master;
        file "/etc/bind/example.net.db";
      };

zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/192.168.0.db";
      };
```

← For forward lookup (Hostname → IP).

← For reverse lookup (IP → hostname).

Zone File for Forward Lookup

/etc/bind/example.net.db (The file name is specified in `named.conf`)

```
$TTL 3D ; default expiration time of all resource records without
        :   their own TTL
@       IN      SOA     ns.example.net. admin.example.net. (
        1               ; Serial
        8H              ; Refresh
        2H              ; Retry
        4W              ; Expire
        1D )            ; Minimum

@       IN      NS      ns.example.net.         ;Address of nameserver
@       IN      MX      10 mail.example.net.    ;Primary Mail Exchanger

www     IN      A       192.168.0.101   ;Address of www.example.net
mail    IN      A       192.168.0.102   ;Address of mail.example.net
ns      IN      A       192.168.0.10    ;Address of ns.example.net
*.example.net. IN A     192.168.0.100   ;Address for other URL in
                                        ;  the example.net domain
```

@: Represents the origin specified in `named.conf` (string after "zone") [`example.net`]

Zone File for Reverse Lookup

`/etc/bind/192.168.0.db`: (The file name is specified in `named.conf`)

```
$TTL 3D
@       IN      SOA     ns.example.net. admin.example.net. (
                        1
                        8H
                        2H
                        4W
                        1D)
@       IN      NS      ns.example.net.

101     IN      PTR     www.example.net.
102     IN      PTR     mail.example.net.
10      IN      PTR     ns.example.net.
```

Testing the setup

```
$ dig www.example.net
<<>> DiG 9.5.0b2 <<>> www.example.net
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27136
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1,
    ADDITIONAL: 1

;; QUESTION SECTION:
;www.example.net.                IN      A

;; ANSWER SECTION:
www.example.net.         259200  IN      A       192.168.0.101

;; AUTHORITY SECTION:
example.net.             259200  IN      NS      ns.example.net.

;; ADDITIONAL SECTION:
ns.example.net.          259200  IN      A       192.168.0.10
```

Need to ensure that Resolv.conf is pointing to the recently setup DNS server

DNS Queries

Message formats are defined in RFC 1035.

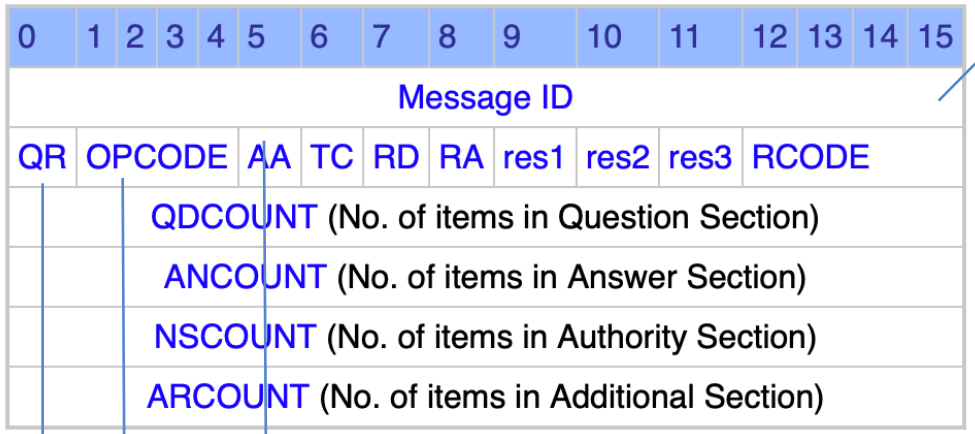
The good news is that each message has the same generic format with 5 sections. This is the last good news.

Section	Meaning/Use
Section 1	Message Header
Section 2	The DNS question being asked (aka Question Section)
Section 3	The Resource Record(s) which answer the question (aka Answer Section)
Section 4	The Resource Record(s) which point to the domain authority (aka Authority Section)
Section 5	The Resource Record(s) which may hold additional information (aka Additional Section)

http://www.zytrax.com/books/dns/ch15/

DNS Query Format

Message Header



Sent in the query and reflected back by the response

Authoritative Answer

0: query, 1: inverse query; 2: status

QR=0 query; QR=1 response

http://www.zytrax.com/books/dns/ch15/

DNS Question Section

Question Section

```
// assume an A query with a name of www.mydomain.com
// the hex representation is
03 77 77 77 08 6D 79 64 6F 6D 61 69 6E 03 63 6F 6D 00
// printable
 !  w  w  w  !  m  y  d  o  m  a  i  n  !  c  o  m  !
```

Field Name	Meaning/Use
QNAME	The domain name being queried
QTYPE	The resource records being requested
QCLASS	The Resource Record(s) class being requested, for instance, inter

Value	Meaning/Use
x'0001 (1)	Requests the A record for the domain name
x'0002 (2)	Requests the NS record(s) for the domain name
x'0005 (5)	Requests the CNAME record(s) for the domain name
x'0006 (6)	Requests the SOA record(s) for the domain name
x'000B (11)	Requests the WKS record(s) for the domain name
x'000C (12)	Requests the PTR record(s) for the domain name
x'000F (15)	Requests the MX record(s) for the domain name
x'0021 (33)	Requests the SRV record(s) for the domain name
x'001C (28)	Requests the AAAA record(s) for the domain name
x'00FF (255)	Requests ANY resource record (typically wants SOA, MX, NS and MX)

http://www.zytrax.com/books/dns/ch15/

DNS Question Section

Answer Section

Field Name	Meaning/Use
NAME	The name being returned e.g. www or ns1.example.net If the name is in the same domain as the question then typically only the host part (label) is returned, if not then a FQDN is returned.
TYPE	The RR type, for example, SOA or AAAA
CLASS	The RR class, for instance, Internet, Chaos etc.
TTL	The TTL in seconds of the RR, say, 2800
RLENGTH	The length of RR specific data in octets, for example, 27
RDATA	The RR specific data (see Binary RR Formats below) whose length is defined by RDLENGTH, for instance, 192.168.254.2

http://www.zytrax.com/books/dns/ch15/

DNS Question Section

Authority Section

Field Name	Meaning/Use
NAME	The name being returned e.g. www or ns1.example.net If the name is in the same domain as the question then typically only the host part (label) is returned, if not then a FQDN is returned.
TYPE	The RR type, for example, SOA or AAAA
CLASS	The RR class, for instance, Internet, Chaos etc.
TTL	The TTL in seconds of the RR, say, 2800
RLENGTH	The length of RR specific data in octets, for example, 27
RDATA	The RR specific data (see Binary RR Formats below) whose length is defined by RDLENGTH, for instance, 192.168.254.2

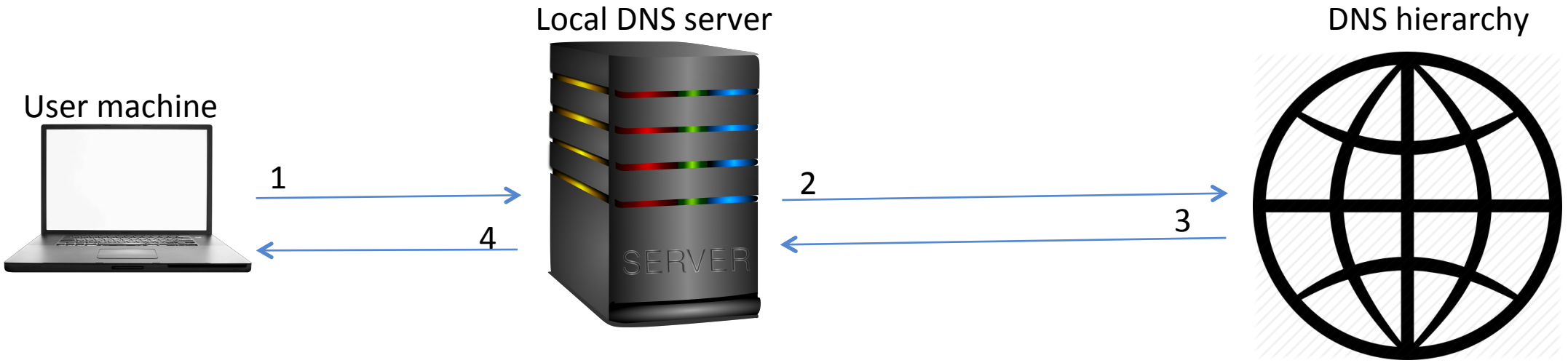
This section mentions the servers that are the ultimate authority for answering DNS queries.

Answers, may be obtained from the cache of other DNS servers.

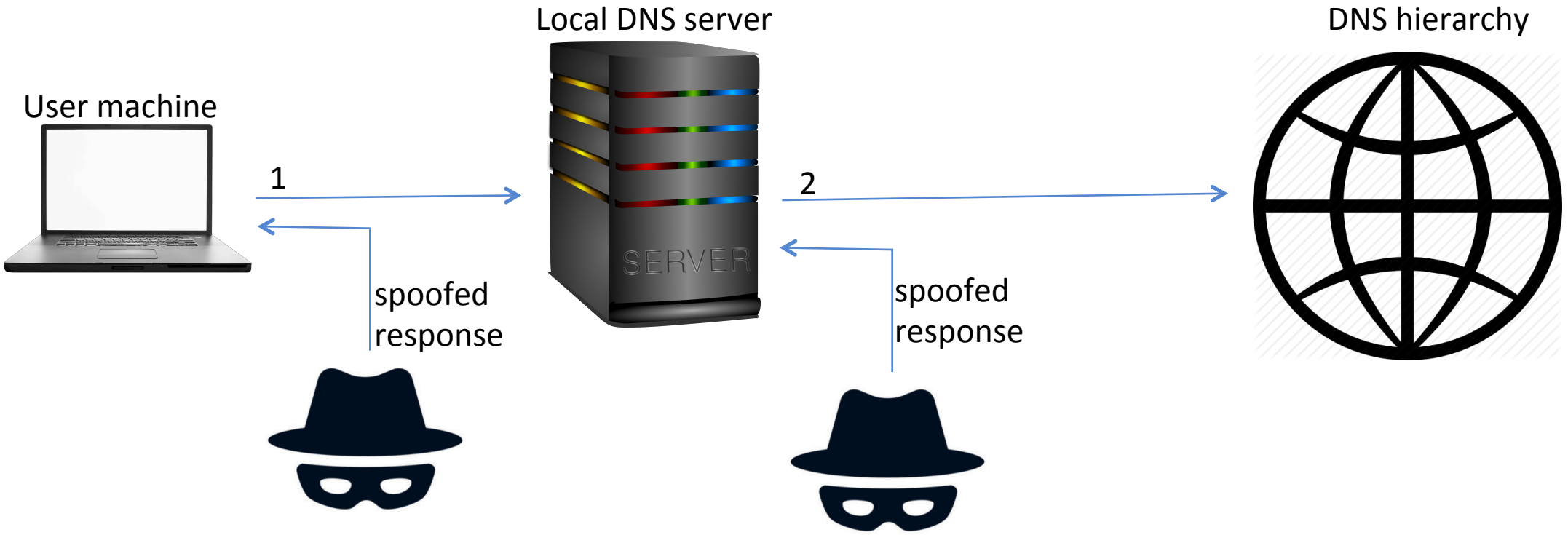
Can be used to check with the authoritative response.

http://www.zytrax.com/books/dns/ch15/

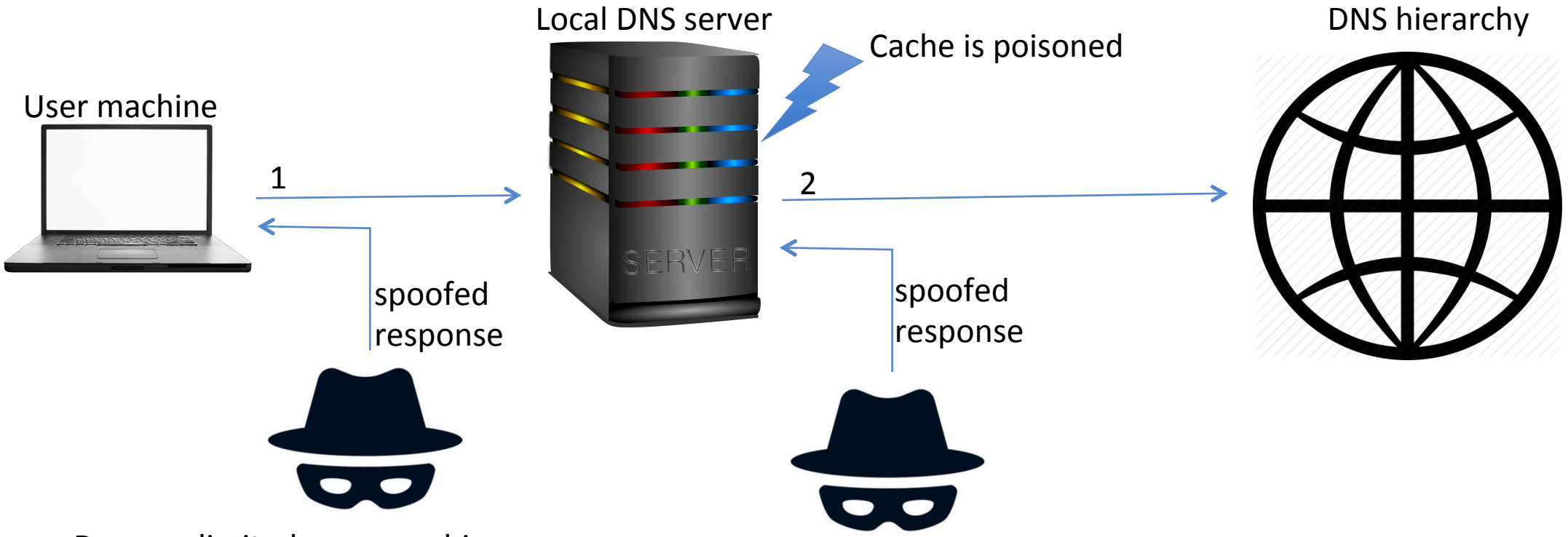
Attacks on Local DNS Servers
Cache Poisoning Attacks



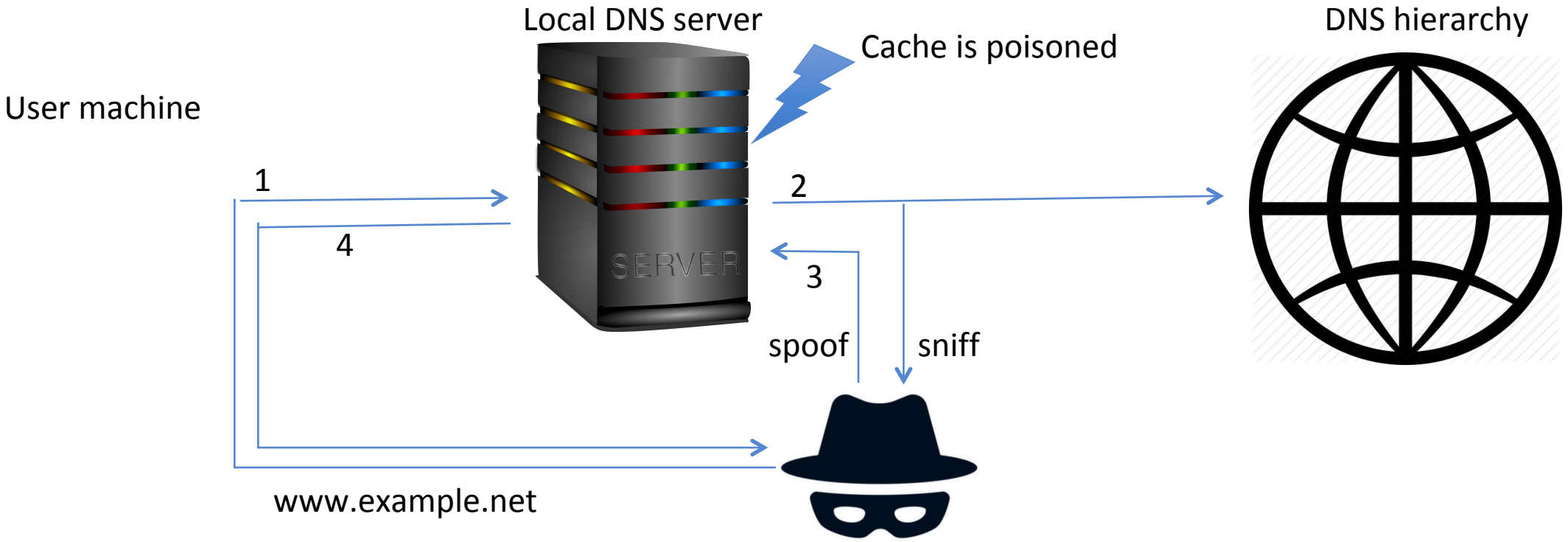
Attacks on Local DNS Servers
Cache Poisoning Attacks



Attacks on Local DNS Servers
Cache Poisoning Attacks



Attacks on Local DNS Servers
Cache Poisoning Attacks



Local DNS Cache Poisoning Attack

Goal: Forge DNS replies after seeing a query from Local DNS Server
Technique: Sniffing and Spoofing

```
Title: Sniff and send DNS answers
    Usage: netwox 105 -h data -H ip -a data -A ip [-d device]
                      [-T uint32] [-f filter] [-s spoofip]
    Parameters:
    -h|--hostname data    hostname
    -H|--hostnameip ip    IP address
    -a|--authns data      authoritative nameserver
    -A|--authnsip ip      authns IP
    -d|--device device    device name
    -T|--ttl uint32       ttl in seconds
    -f|--filter filter    pcap filter
    -s|--spoofip spoofip  IP spoof initialization type
```

Local DNS Cache Poisoning Attack

```
$ sudo netwox 105 --hostname www.example.net
    --hostnameip 10.20.30.40 --authns ns.example.net
    --authnsip 10.20.30.50 --filter src host 10.0.2.16
    --ttl 19000 --spoofip raw
```

Attack

Result

```
$ dig www.example.net

;; QUESTION SECTION:
;www.example.net.                IN      A

;; ANSWER SECTION:
www.example.net.         19000   IN      A       10.20.30.40

;; AUTHORITY SECTION:
.                        19000   IN      NS      ns.example.net.

;; ADDITIONAL SECTION:
ns.example.net.          19000   IN      A       10.20.30.50
```

❶

Inspect the Cache

```
;
$DATE 20151214152854
; authanswer
.                   18997   IN   NS     ns.example.net.
; authauthority
ns.example.net.     18997   NS          ns.example.net.
; additional
                    18997   A           10.20.30.50
; authanswer
www.example.net.    18997   A           10.20.30.40    ← Poisoned!
;
```

- Run "`sudo rndc dumpdb -cache`" and check the contents of "`/var/cache/bind/dump.db`".
- Clean the cache using "`sudo rndc flush`" before doing the attack.

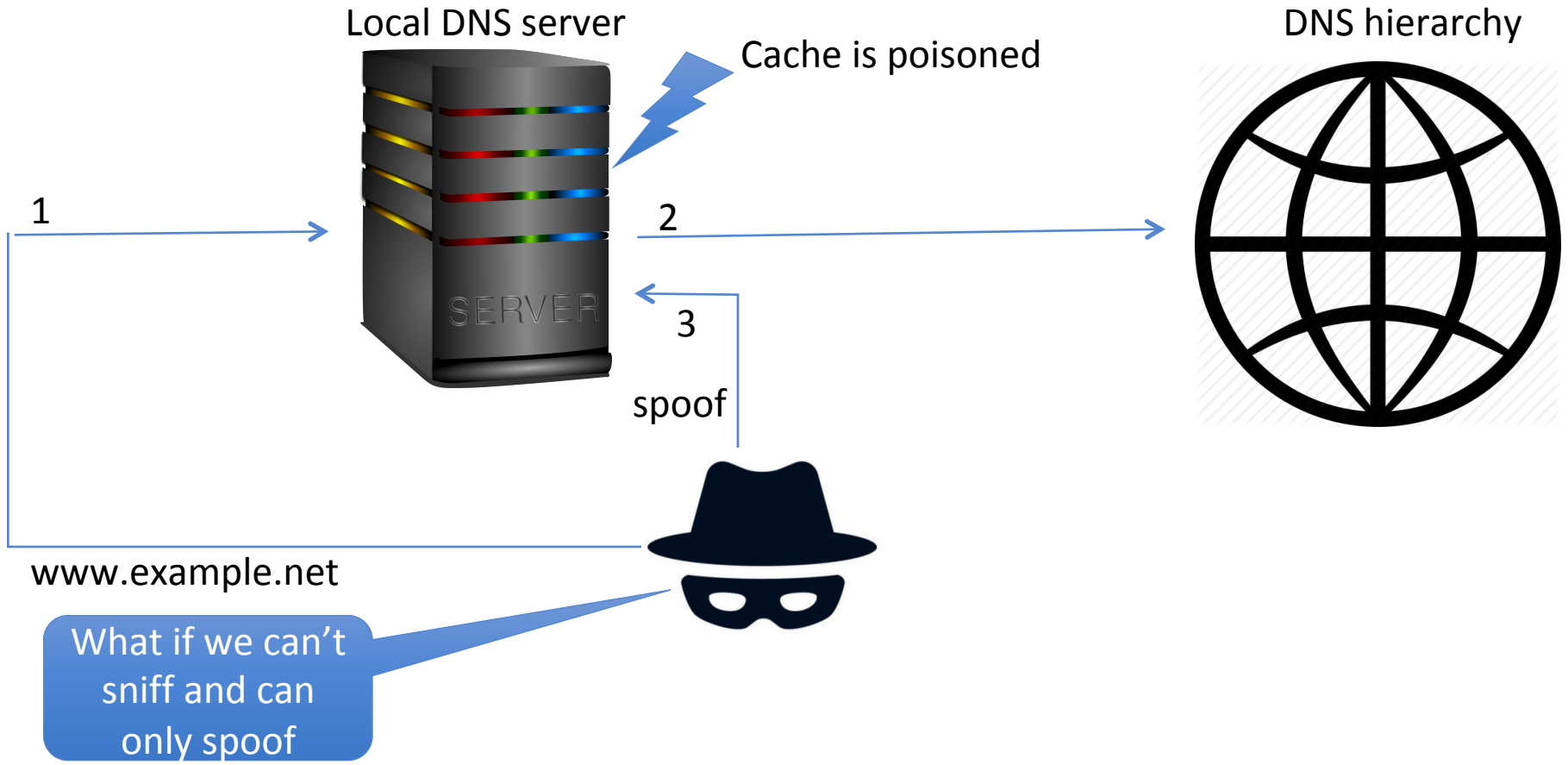
Targeting the Authority section

```
$ sudo netwox 105 --hostname www.example.net
    --hostnameip 10.20.30.40 --authns ns.attacker.net
    --authnsip 10.20.30.50 --filter src host 10.0.2.16
    --ttl 19000 --spoofip raw
```

Can target the authority section

Any DNS query sent to the local DNS server will be (if needed) directed to the attacker's ns.attacker.net

Attacks on Local DNS Servers
Cache Poisoning Attacks



Cache Poisoning Without Sniffing

Section	Meaning/Use
Section 1	Message Header
Section 2	The DNS question being asked (aka Question Sectio
Section 3	The Resource Record(s) which answer the question
Section 4	The Resource Record(s) which point to the domain a
Section 5	The Resource Record(s) which may hold additional i

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Message ID															
QR	OPCODE			AA	TC	RD	RA	res1		res2		res3	RCODE		
QDCOUNT (No. of items in Question Section)															
ANCOUNT (No. of items in Answer Section)															
NSCOUNT (No. of items in Authority Section)															
ARCOUNT (No. of items in Additional Section)															

Two difficulties in creating a valid spoof:

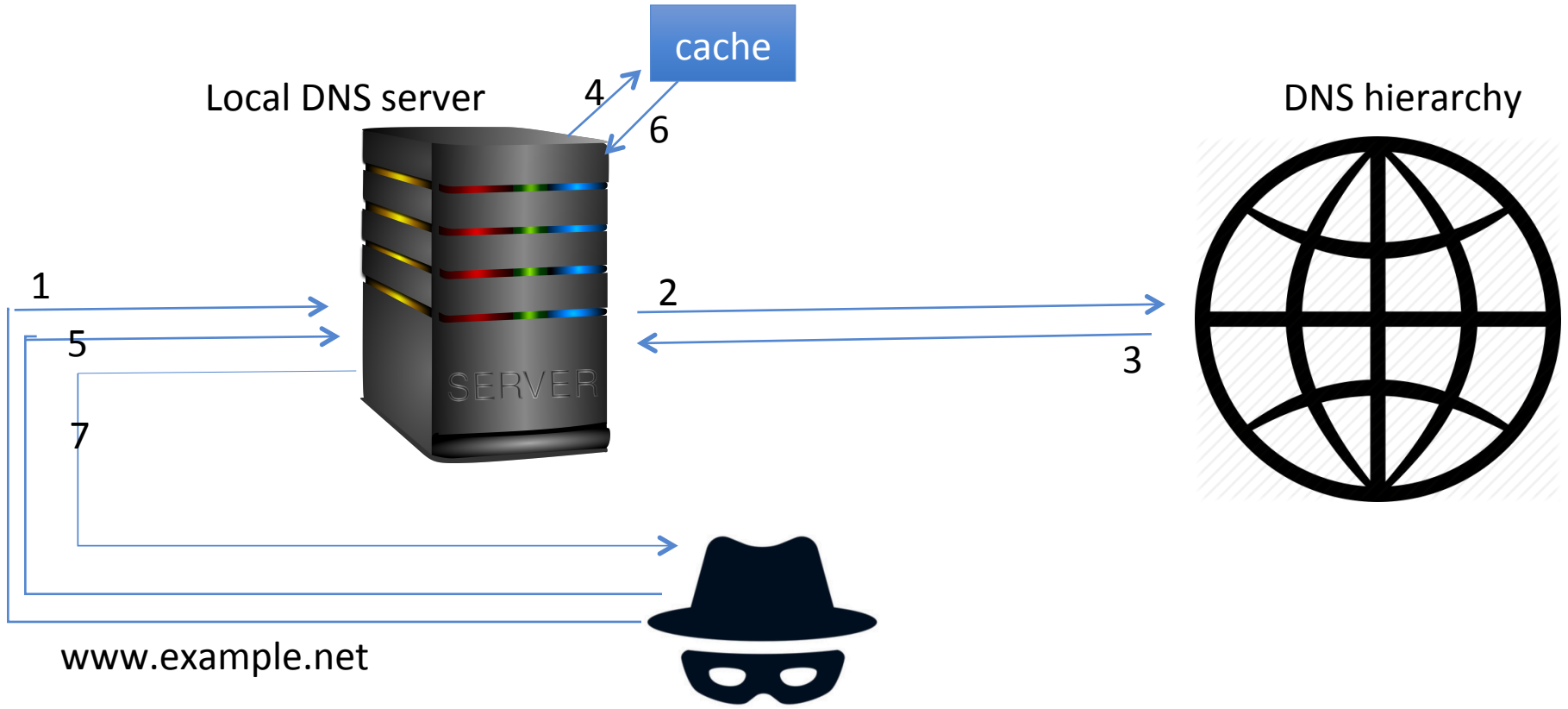
1. Need to guess the local DNS server's source port (2^16 possibilities)
2. The response should have the same Message ID as the DNS query in step (2).

(Brute force attack 2^32 →at 1000 spoofed queries / second, it will take 50 days to try all 2^32 possibilities

Further Difficulties: the Local DNS server's cache

If the real response (3) arrives and it is cached (4). Then subsequent queries will read off the cache (5 →6 →7) and no query is made from the Local DNS.

Thus, to make another try, the attacker should wait till the cache is flushed.



Flaw in the Protocol

- When looking up sibling names like 1.google.com, and 2.google.com.
  - Attackers can do this and say they're the official server for www.google.com, telling the local DNS server what www. needs to be, and the local DNS will believe the attacker.

https://duo.com/blog/the-great-dns-vulnerability-of-2008-by-dan-kaminsky

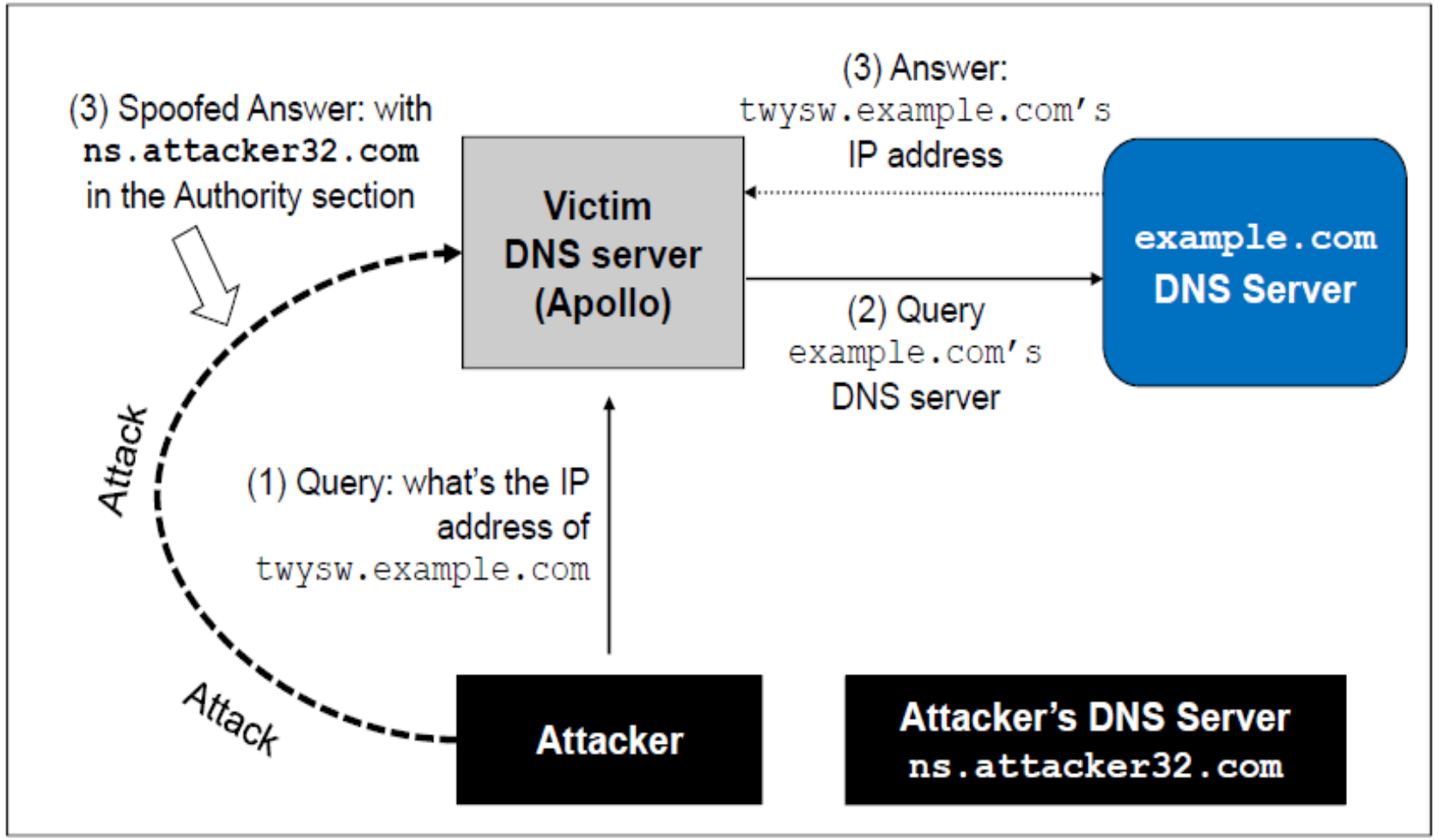
Kaminsky Attack

How to keep trying spoofed DNS responses (2^32 times) without worrying about the cache effect?

Kaminsky's Idea:

- Ask a different question every time, so caching the answer does not matter, and the local DNS server will send out a new query each time.
- Provide forged answer in the Authority section

Kaminsky Attack



The Kaminsky Attack: A Sample Response

This random name will change for each attack attempt

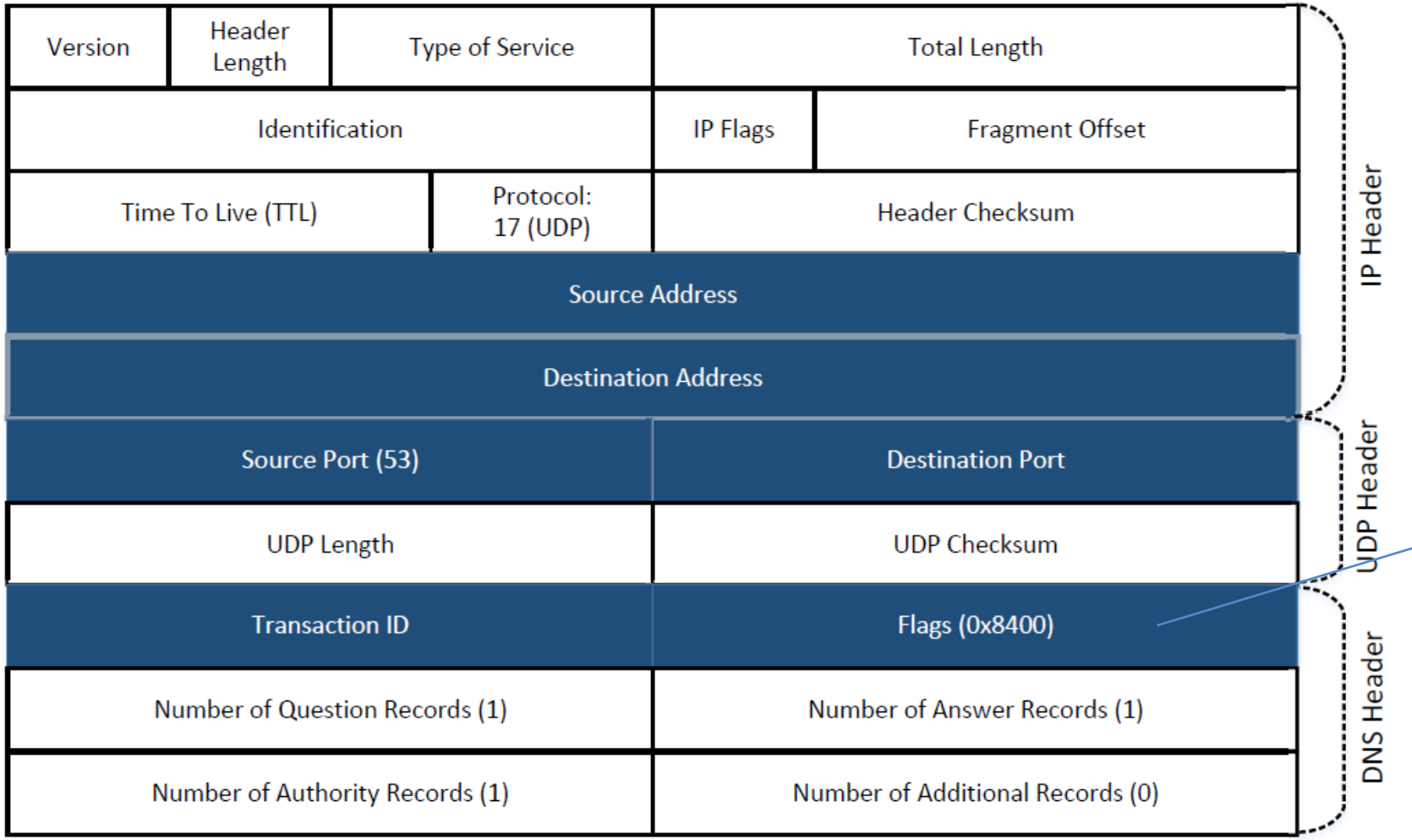
This answer does not matter

```
;; QUESTION SECTION:
;twysw.example.com.              IN    A

;; ANSWER SECTION:
twysw.example.com.      259200  IN    A    1.2.3.4

;; AUTHORITY SECTION:
example.com.            259200  IN    NS   ns.attacker32.com
```

This is what we want the local DNS server to cache

Tell the DNS server to use this one as the nameserver for the example.com domain

Spoofing Replies: IP and UDP headers



DNS Response
Answer is authoritative

Spoofing Replies: DNS Header and Payload

Question Record

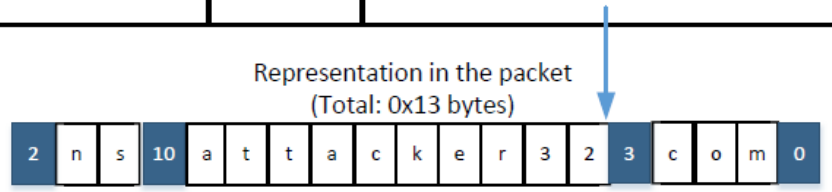
Name	Record Type	Class
twysw.example.com	"A" Record 0x0001	Internet 0x0001

Answer Record

Name	Record Type	Class	Time to Live	Data Length	Data: IP Address
twysw.example.com	"A" Record 0x0001	Internet 0x0001	0x00002000 (seconds)	0x0004	1.2.3.4

Authority Record

Name	Record Type	Class	Time to Live	Data Length	Data: Name Server
example.com	"NS" Record 0x0002	Internet 0x0001	0x00002000 (seconds)	0x0013	ns.attacker32.net



Digital Signatures

Alice

Alice's Private Key

sig_K

y = digital signature

Message

x = "Attack at Dawn!!"

(x, y)

unsecure channel

ver_K

Alice's Public Key

TRUE / FALSE

Everyone Else

Signing Function

$y = sig_a(x)$

Input : Message (x) and Alice's private key
Output: Digital Signature of Message

Verifying Function

$ver_b(x, y)$

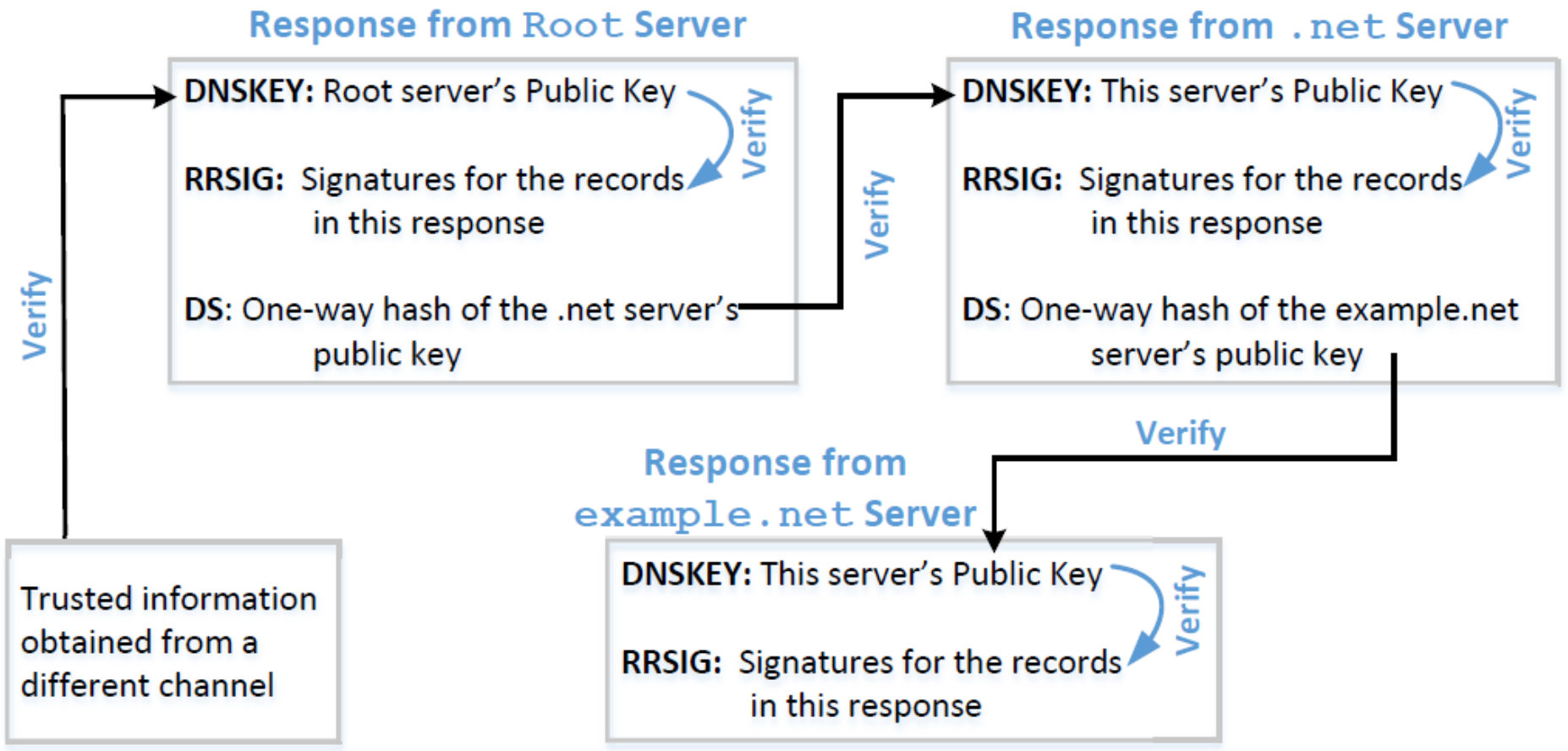
Input : digital signature, message
Output : true or false
true if signature valid
false otherwise

Protection Against DNS Cache Poisoning Attacks

DNSSEC

- DNSSEC is a set of extension to DNS, aiming to provide authentication and integrity checking on DNS data.
- With DNSSEC, all answers from DNSSEC protected zones are digitally signed.
- By checking the digital signatures, a DNS resolver is able to check if the information is authentic or not.
- DNS cache poisoning will be defeated by this mechanism as any fake data will be detected because they will fail the signature checking.

Protection Using DNSSEC



Protection Using TLS/SSL

Transport Layer Security (TLS/SSL) protocol provides a solution against the cache poisoning attacks.

- After getting the IP address for a domain name (www.example.net) using DNS protocol, a computer will ask the owner (server) of the IP address to prove that it is indeed www.example.net.
- The server has to present a public-key certificate signed by a trusted entity and demonstrates that it knows the corresponding private key associated with www.example.net (i.e., it is the owner of the certificate).
- HTTPS is built on top of TLS/SSL. It defeats DNS cache poisoning attacks.

DNSSEC versus TLS/SSL

- Both DNSSEC and TLS/SSL are based on the public key technology, but their chains of trust are different.
- DNSSEC provides chain of trust using DNS zone hierarchy, so nameservers in the parent zones vouch for those in the child zones.
- TLS/SSL relies on Public Key Infrastructure which contains Certificate Authorities vouching for other computers.

Denial of Service Attacks on Root Servers

Attacks on the Root and TLD Servers :

Root nameservers: If the attackers can bring down the servers of the root zone, they can bring down the entire Internet. However, attack root servers is difficult:

- The root nameservers are highly distributed. There are 13 (A,B….M) root nameservers (server farm) consisting of a large number of redundant computers to provide reliable services.
- As the nameservers for the TLDs are usually cached in the local DNS servers, the root servers need not be queried till the cache expires (48 hrs). Attacks on the root servers must last long to see a significant effect.

Denial of Service Attacks on TLD Servers

Nameservers for the TLDs are easier to attack. TLDs such as gov, com, net etc have quite resilient infrastructure against DOS attacks. But certain obscure TLDs like country-code TLDs do not have sufficient infrastructure. Due to this, the attackers can bring down the Internet of a targeted country.

Attacks on Nameservers of a Particular Domain

Dyn network : In 2016, multiple DDoS attacks were launched against a major DNS service provider for companies like CNN, BBC, HBO, PayPal etc. The attacks are believed to have been launched through botnet consisting of different IoT devices like IP cameras, baby monitors etc. It caused major Internet services unavailable .

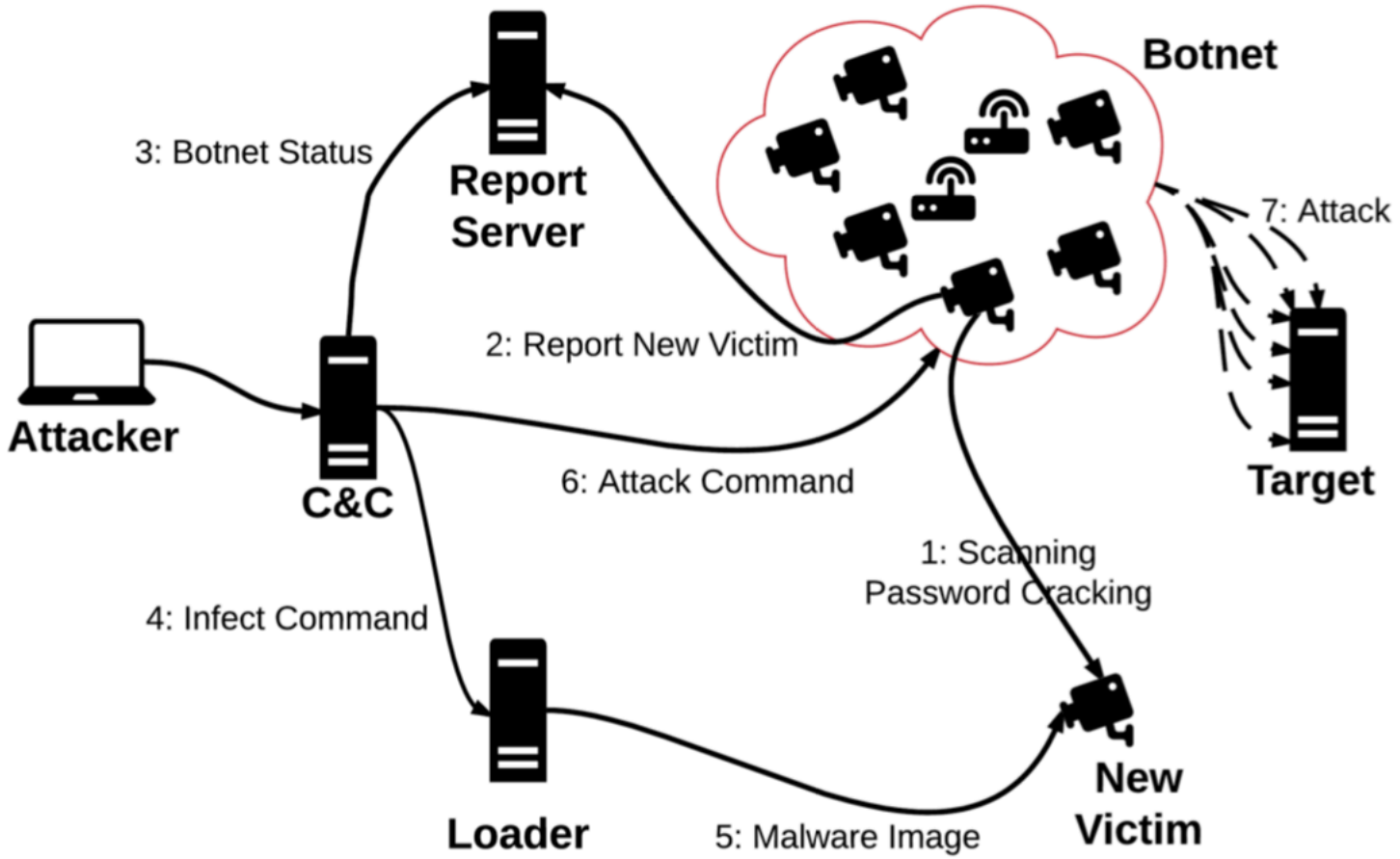
Mirai Botnet Malware

- Number of IoT devices increasing at a rapid rate
- These devices are characterized by
  - Low profile
  - Less user interactions
  - Security often compromised (for better performance / smaller profile)
  - Not always up-to-date with security patches
- Malware with IoT devices as targets
  - Bashlight and Mirai are the most popular
  - PNScan, targets x86 platforms.
    - Try to determine router login baed on a special dictionary
    - Connect using ssh connection using predefined user credentials

Low hanging fruit for hackers

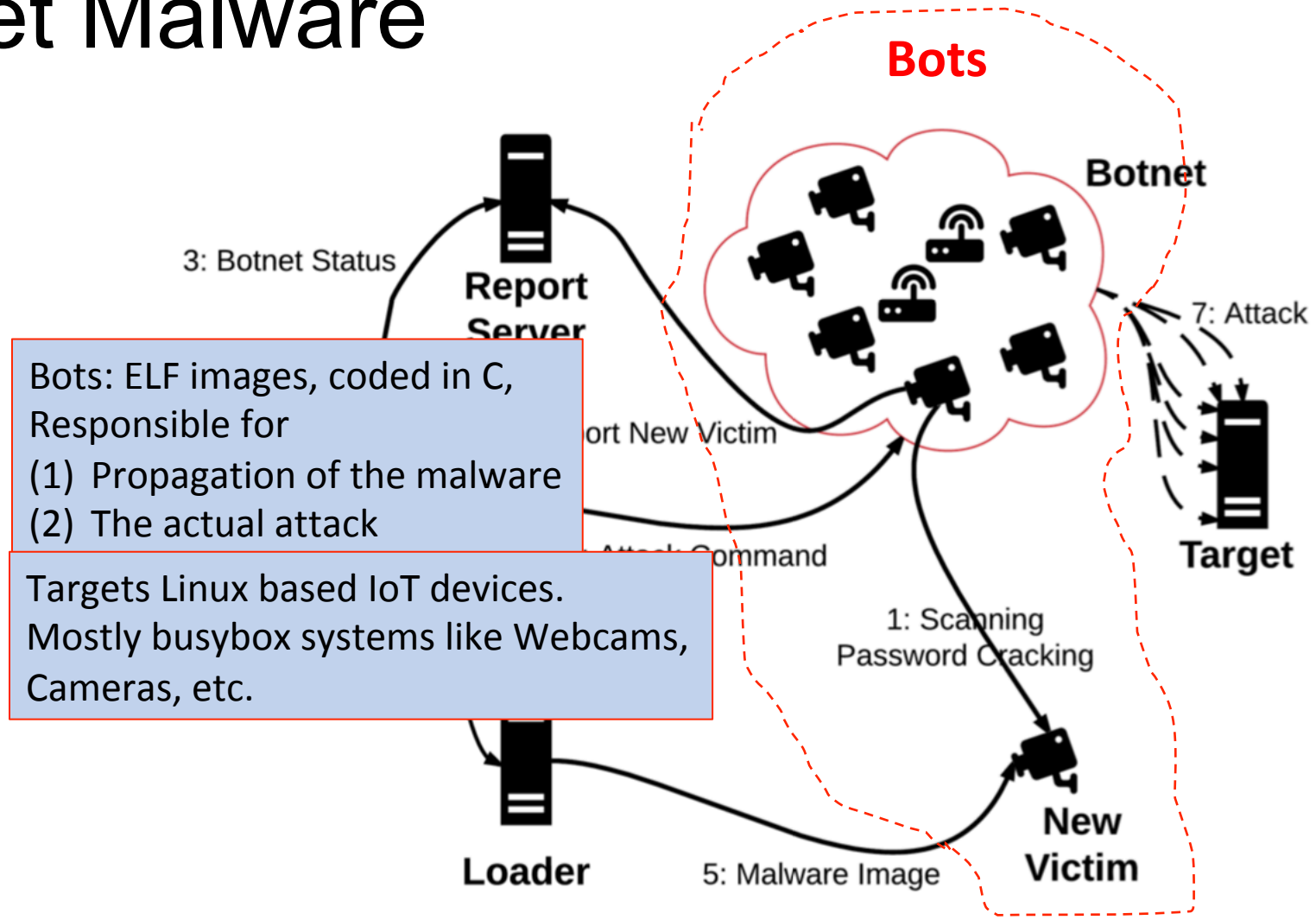
Mirai BotNet Malware

280 Gbps max flooding
50,000 unique Ips
164 countries



The Mirai Botnet and the IoT Zombie Armies, 2017

Mirai BotNet Malware



Bots: ELF images, coded in C,
Responsible for
(1) Propagation of the malware
(2) The actual attack

Targets Linux based IoT devices.
Mostly busybox systems like Webcams,
Cameras, etc.

The Mirai Botnet and the IoT Zombie Armies, 2017

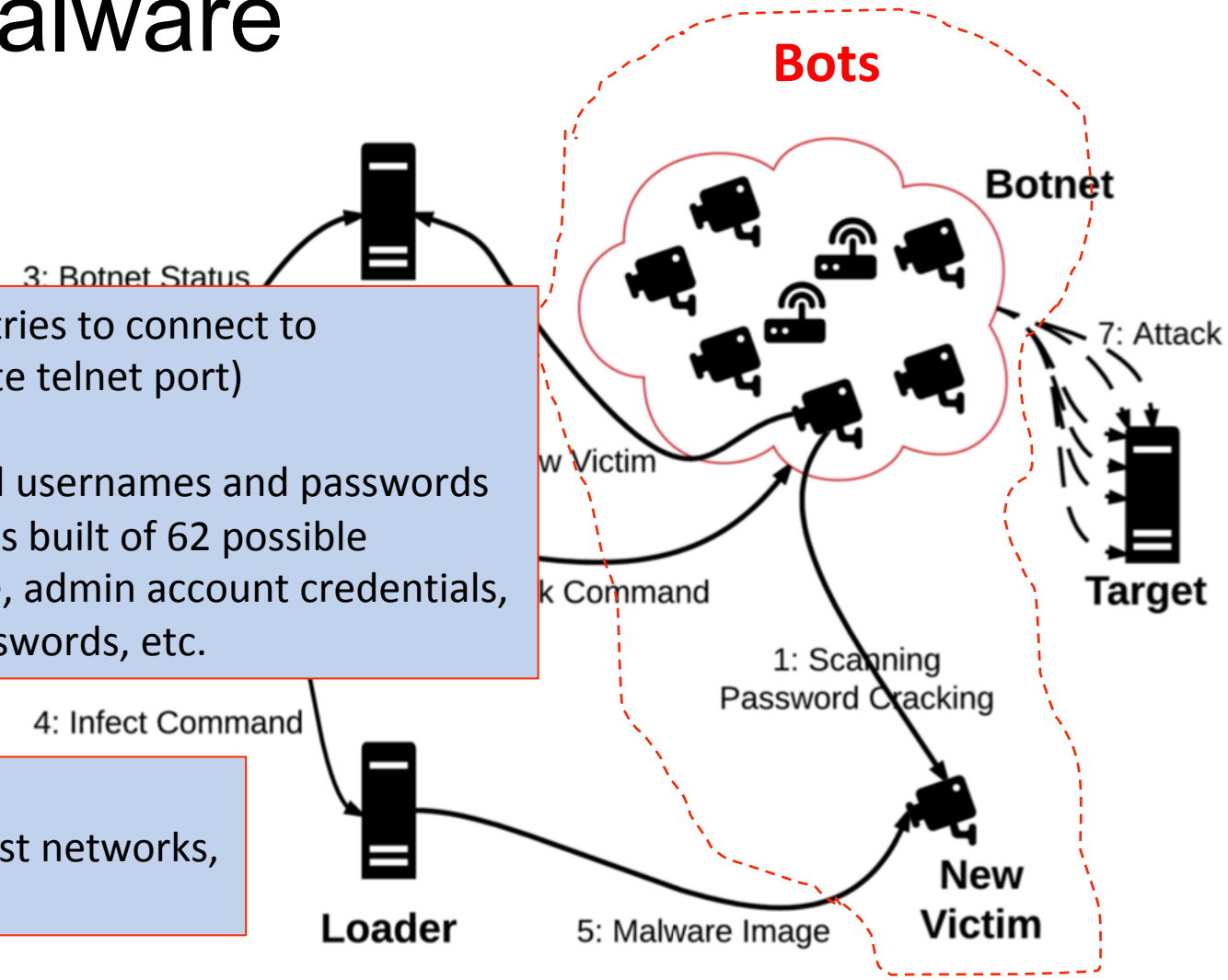
Mirai BotNet Malware

Infiltration

Every bot generates random IPs and tries to connect to Port 23 (telnet) or port 2323 (alternate telnet port)

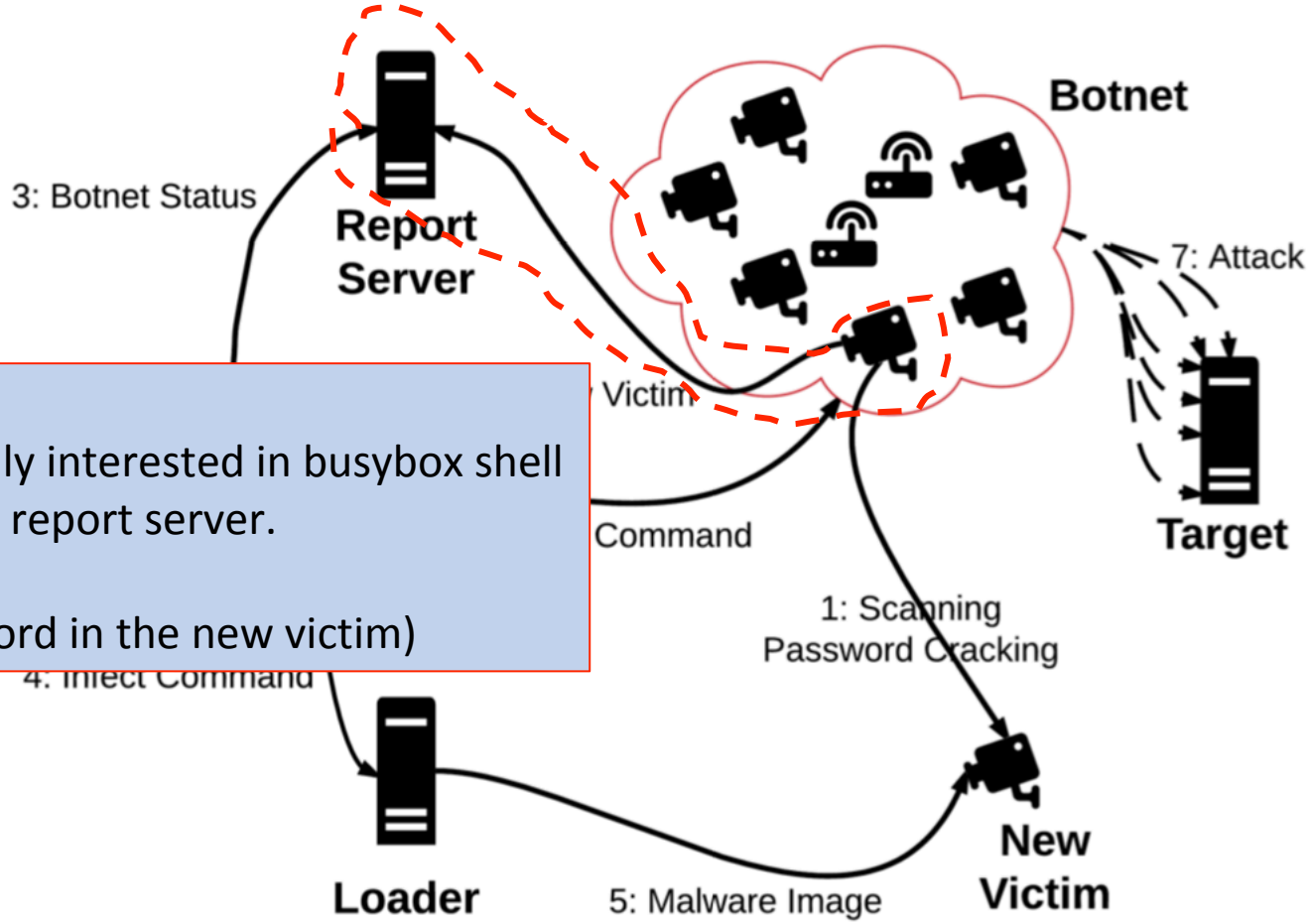
Brute force dictionary search for valid usernames and passwords that will permit login. The dictionary is built of 62 possible username / passwords. These include, admin account credentials, debug logins, usernames with no passwords, etc.

Some IP addresses are blacklisted.
Loopback, internal networks, multicast networks, US postal servcie, DoD, GE, HP, IANA,



The Mirai Botnet and the IoT Zombie Armies, 2017

Mirai BotNet Malware



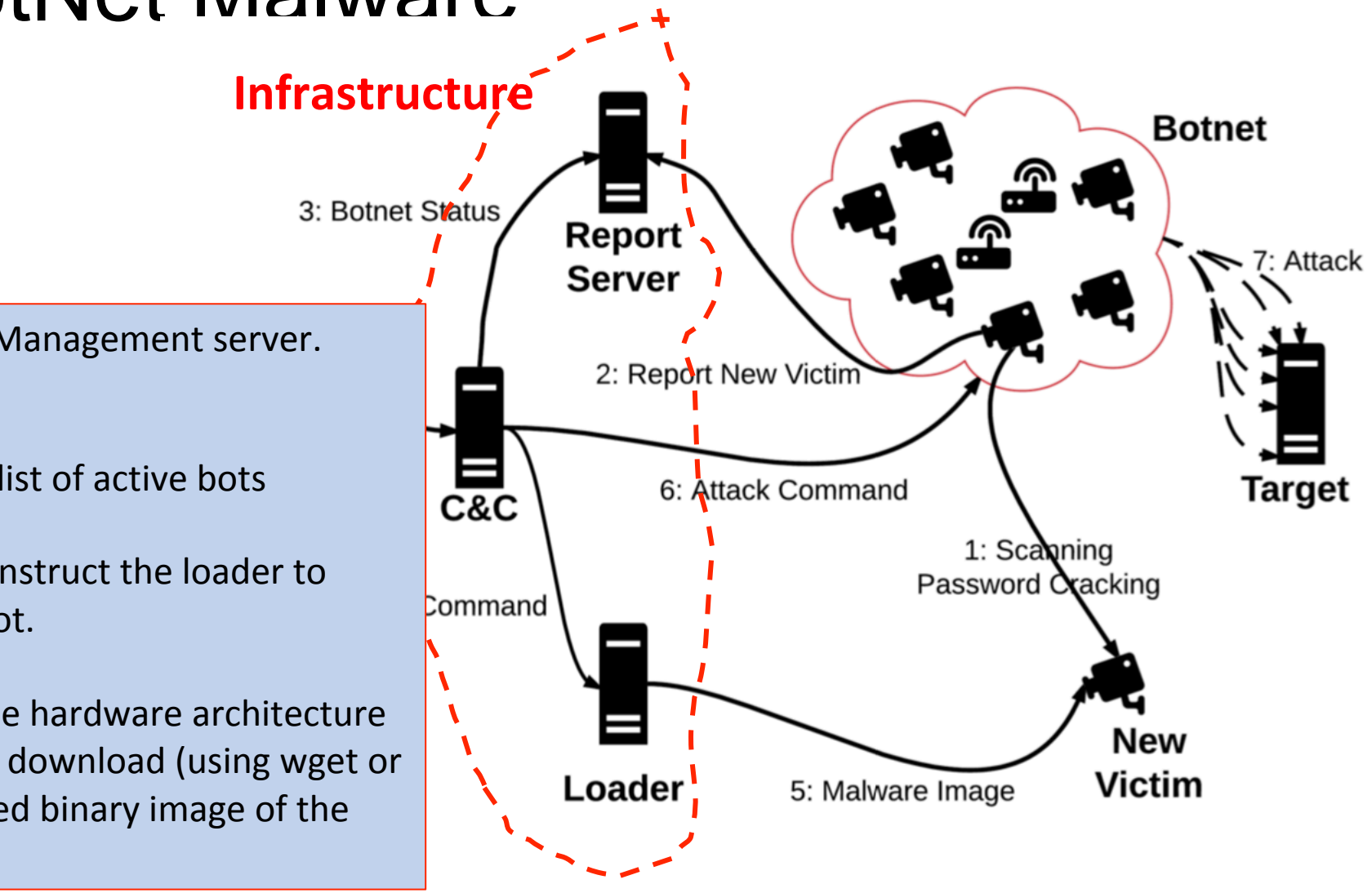
On a successful login:

- Try to establish a shell. Especially interested in busybox shell
- Report a successful login to the report server.

(does not try to change the password in the new victim)

The Mirai Botnet and the IoT Zombie Armies, 2017

Mirai BotNet Malware



Command and Control. Management server. Implemented in Go.

At any time, it can get a list of active bots from the report server.
It can, also, at anytime, instruct the loader to load malware into the bot.

Loader, depending on the hardware architecture of the bot, instructs it to download (using wget or tftp) and execute required binary image of the malware.

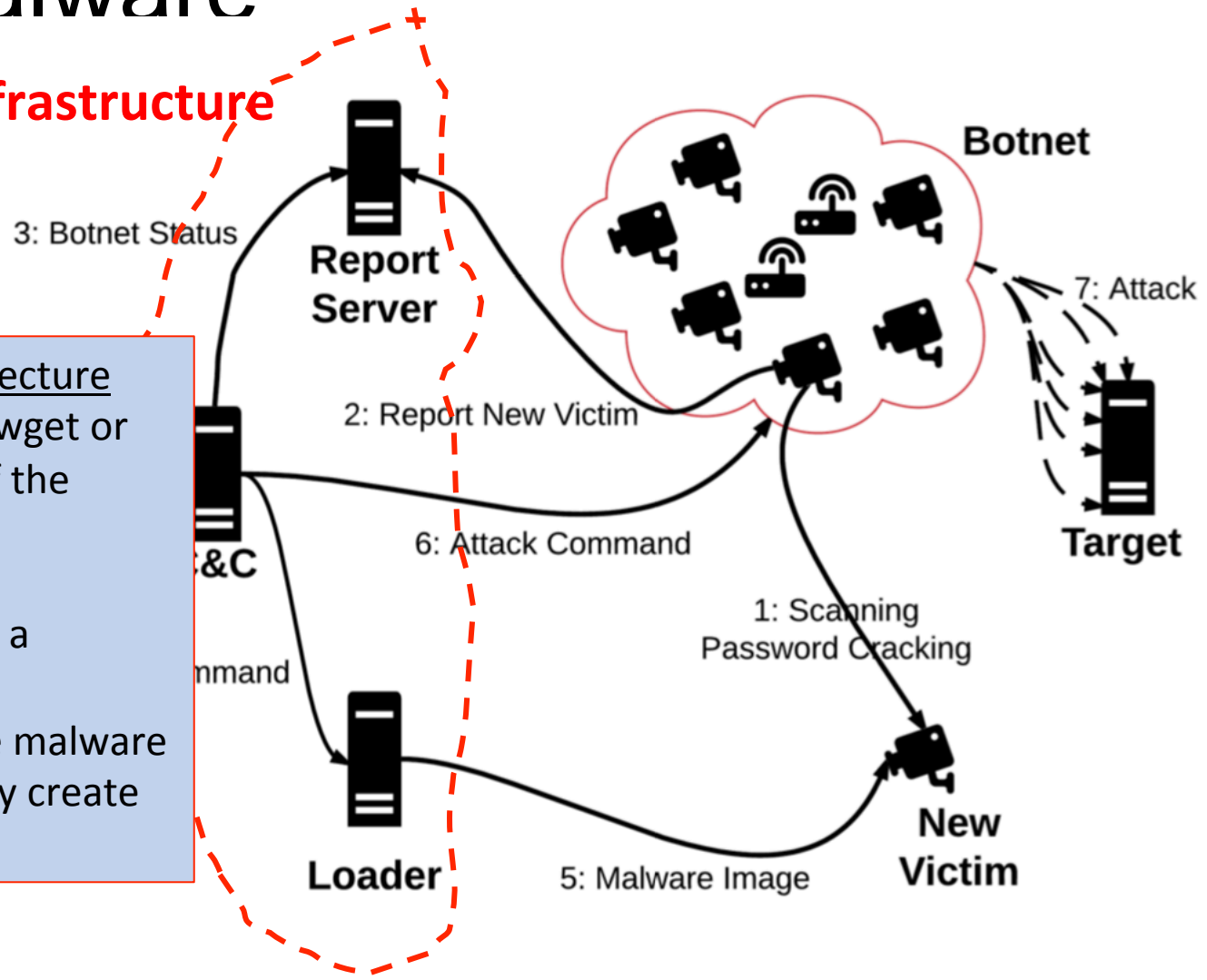
The Mirai Botnet and the IoT Zombie Armies, 2017

Mirai BotNet Malware

18 hardware variants supported including ARM, MIPS, x86, SPARC

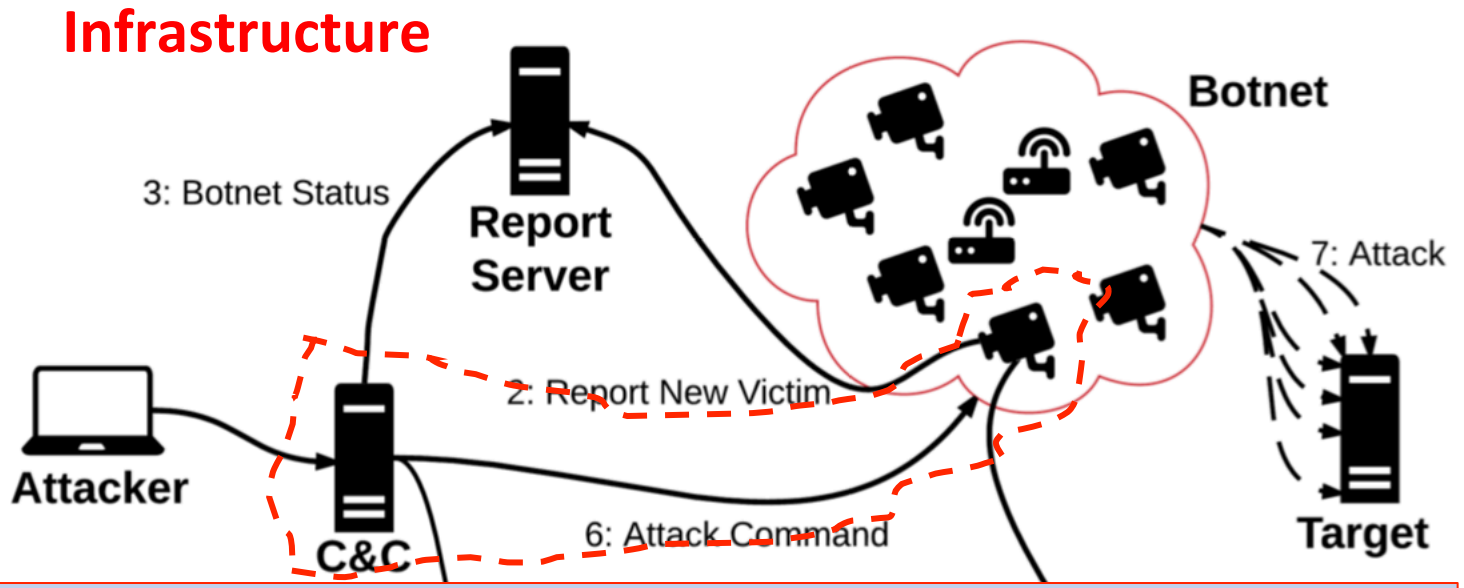
Loader, depending on the hardware architecture of the bot, instructs it to download (using wget or tftp) and execute required binary image of the malware.

Before this is done, the bot needs to know a partition that is writeable.
If tftp or wget clients are not available, the malware will employ *echo* commands to dynamically create the executable.



The Mirai Botnet and the IoT Zombie Armies, 2017

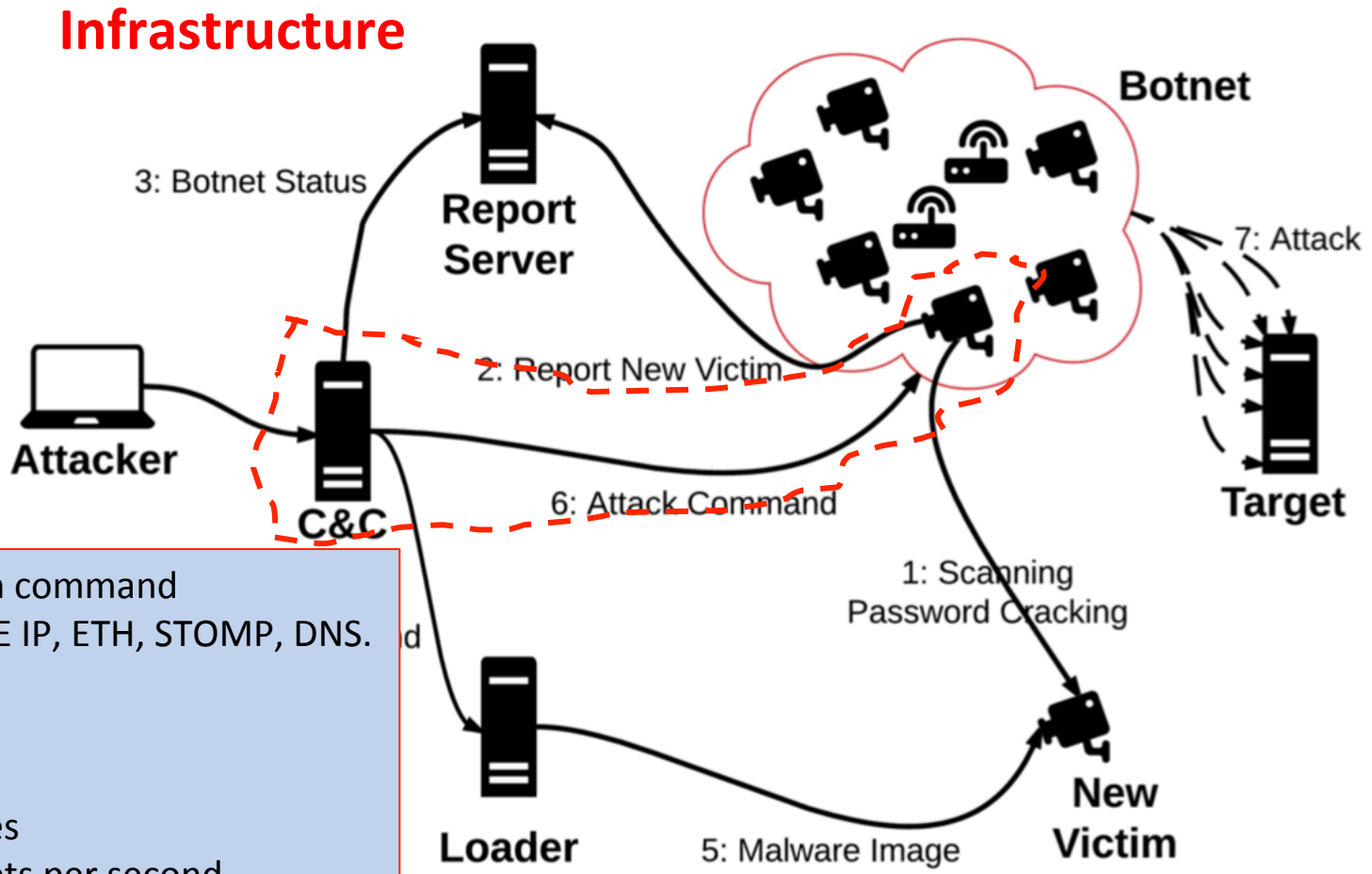
Mirai BotNet Malware



Newly formed bot establishes connection with C&C. Periodic heartbeats between the two.

Other activities in the bot:
* memory scraping to identify other malware present in the bot. If found, kill the process.
  (wants to be the own the device)
- Deletes itself from the persistent storage. Will only be available in RAM (fileless malware)
- Monitors the watchdog timer to defend against system hangs and reboots.
- On command, can start a variety of floods: SYN, ACK, UDP, GRE, DNS, STOMP, ETH. Application layer flooding. 25,000 SYN packets per second.

Mirai BotNet Malware



All botnets attack the target on command
by flooding SYN, ACK, UDP, GRE IP, ETH, STOMP, DNS.
Application level flooding.

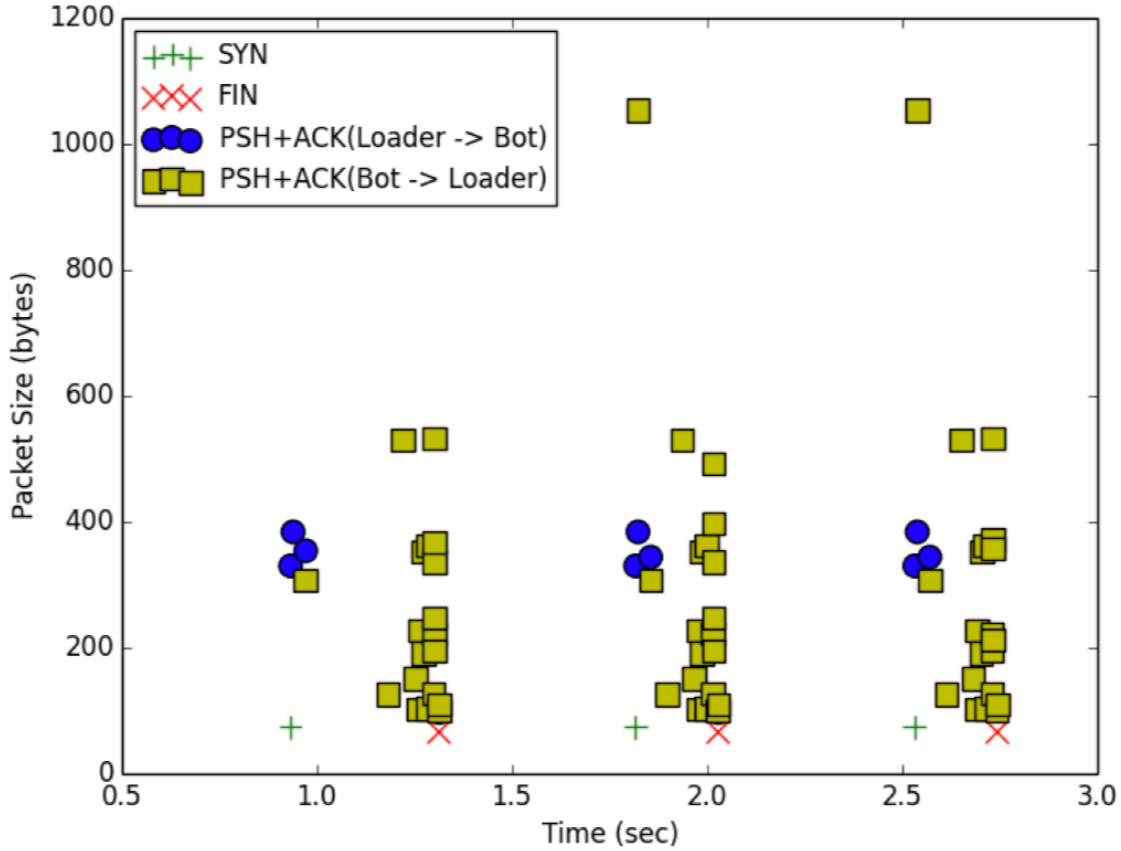
Peak: 620 Gbps
Controls: 0.5 million IOT devices
On raspberry Pi 3: 25000 packets per second

Other Mirai Variants

- US university (Feb 2017)
- Windows based strain
- Mirai strain used for bitcoin mining

Detection

- Patterns of communications
  - Huge communication on ports 23, 2323, 22 for authorization purposes
  - Frequent exchange of traffic with infrastructure
  - Surge of egress traffic throughout the course of the attack



Mitigation

Short-term mitigations

- Blocking TCP ports used for probing and bruteforcing the device
- Filtering egress and ingress packets by TCP rules

Better design strategies

- Least privilege implementations
- capability based systems
- Microkernel /Unikernel based approaches (Linux is too large for embedded applications)