Kaminsky Attack

Revisit DNS Cache Poisoning Attack

Rax

2021/12/26

Domain Name System (DNS)

- translate domain name to IP address

```
vagrant@mooncake:~/demo$ dig girls.hitcon.org

; <<>> DiG 9.11.3-1ubuntu1.16-Ubuntu <<>> girls.hitcon.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19206
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;girls.hitcon.org.              IN      A

;; ANSWER SECTION:
girls.hitcon.org.       300     IN      A       172.67.200.105
girls.hitcon.org.       300     IN      A       104.21.85.2
```

Figure 1

DNS Query Process

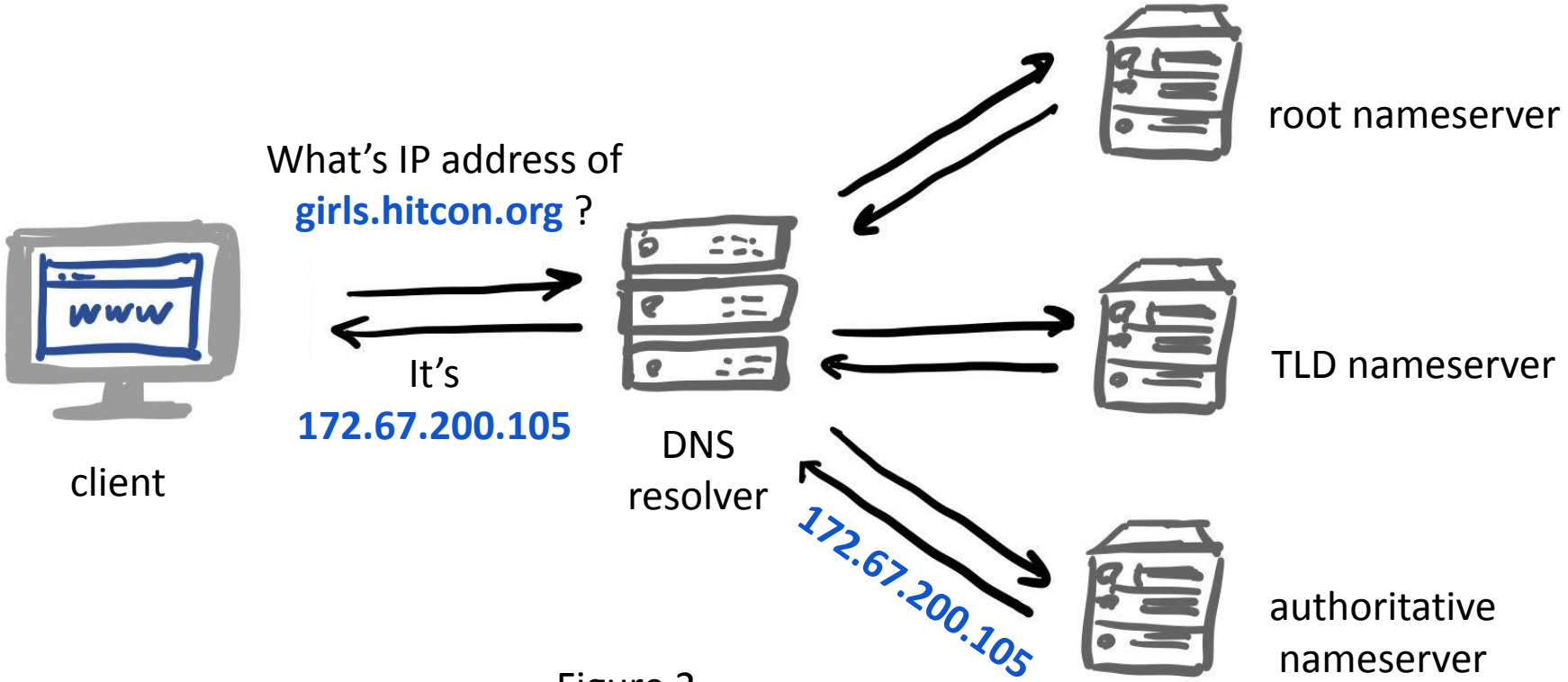


Figure 2

DNS Attacks

- 2 types
  - Denial-of-Service (DoS) attack
  - DNS spoofing attack
    - provide a fake IP address to victims
    - e.g. DNS cache poisoning

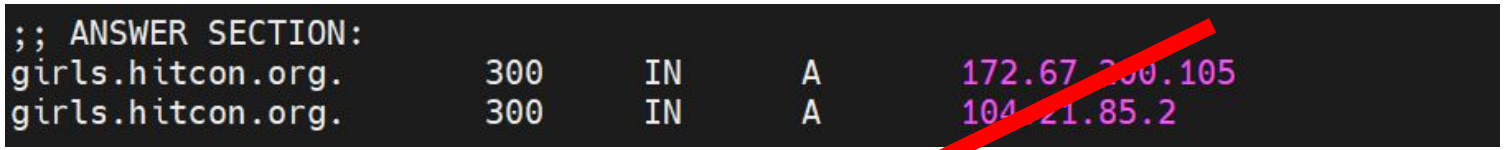


Figure 3

1.2.3.4

(malicious IP address)

DNS Cache Poisoning - Why

- flaw
  - use UDP instead of TCP
  - RFC 8085: “... applications MUST implement corresponding checks at the application layer or explicitly request that the operating system filter the received packets”

DNS Cache Poisoning - What

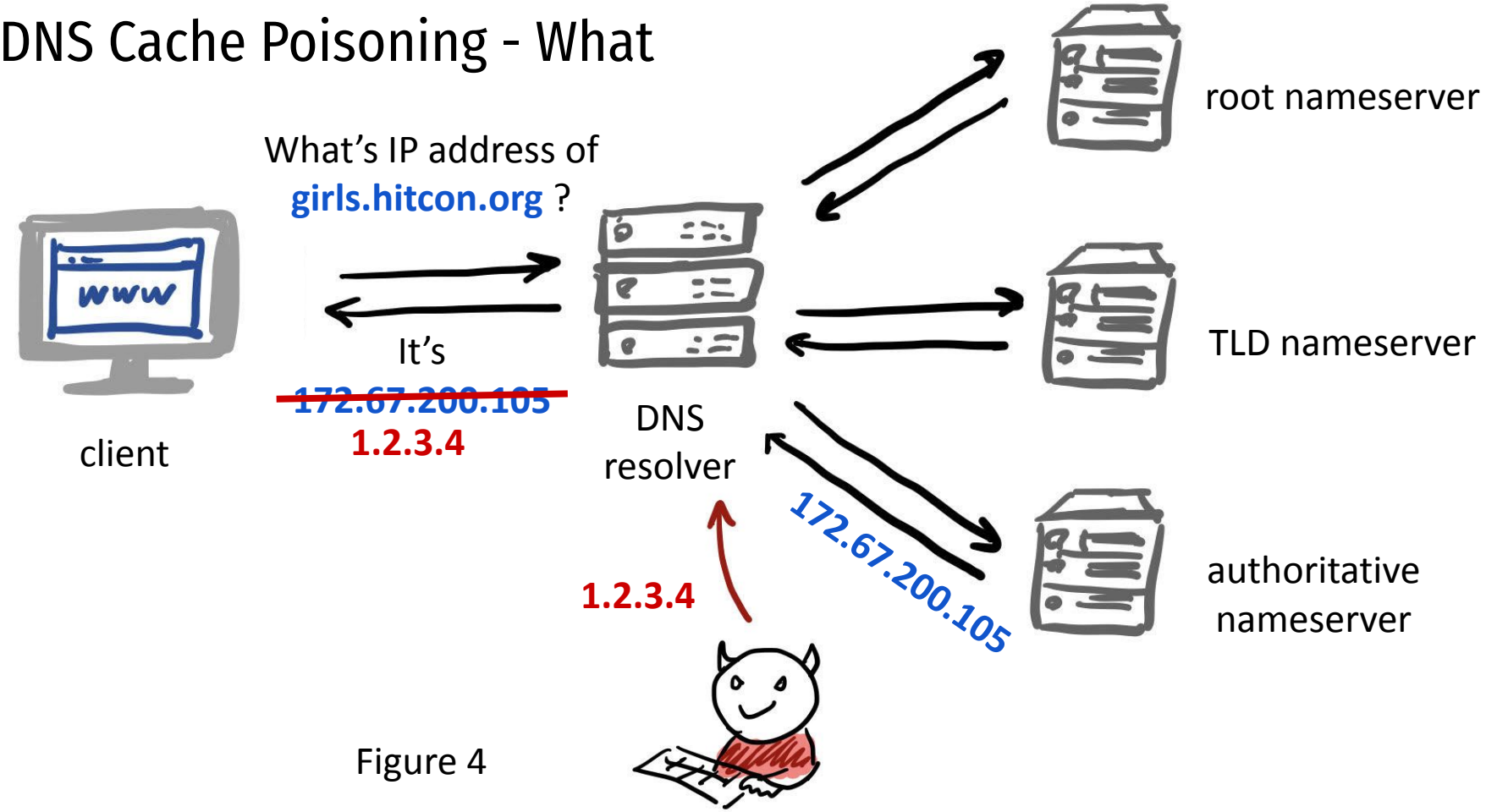


Figure 4

DNS Cache Poisoning - How

- Which cache?
  - resolver’s cache
- How to poison the cache?
  - Whose response to spoof? Who provides the last resolution?
    - authoritative server
  - Spoofed packet?
    - DNS request-response matching

DNS Cache Poisoning - How

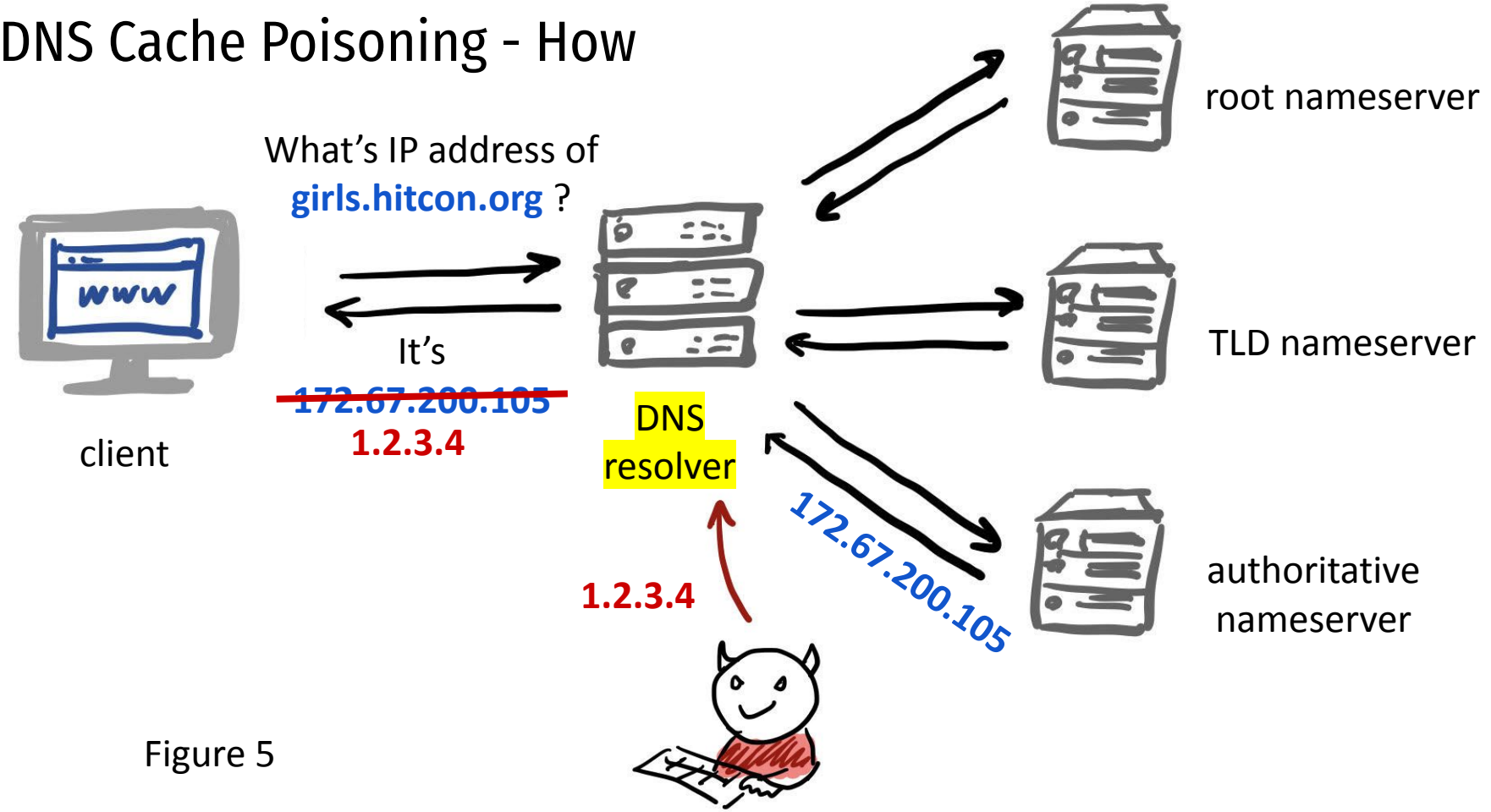


Figure 5

DNS Cache Poisoning - How

- Which cache?
  - resolver’s cache
- How to poison the cache?
  - Whose response to spoof? Who provides the last resolution?
    - authoritative server
  - Spoofed packet?
    - DNS request-response matching

DNS Cache Poisoning - How

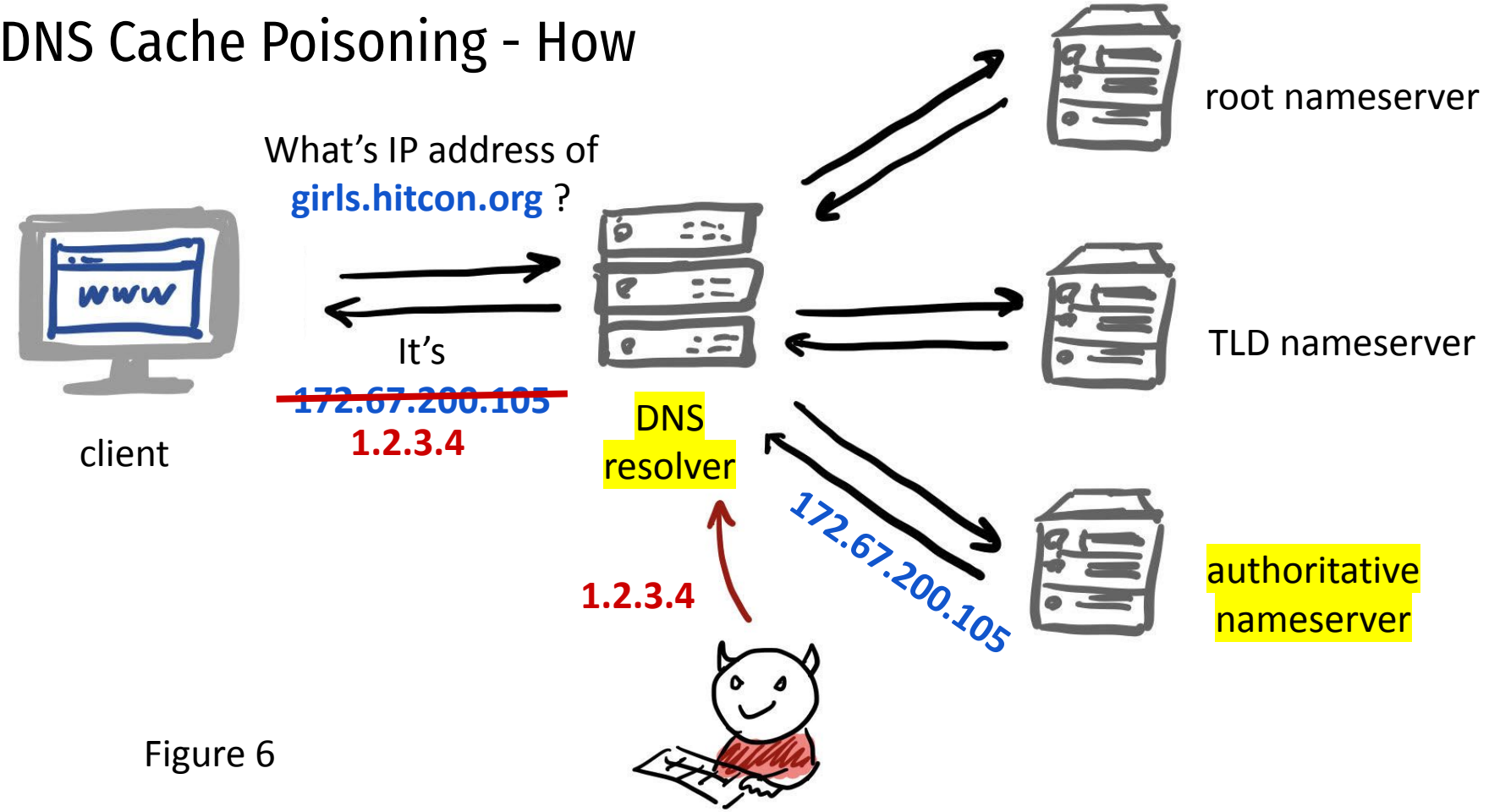


Figure 6

DNS Cache Poisoning - How

- Which cache?
  - resolver’s cache
- How to poison the cache?
  - Whose response to spoof? Who provides the last resolution?
    - authoritative server
  - Spoofed packet?
    - DNS request-response matching

DNS Request-Response Matching

- off-path attack
  - cannot sniff between resolver and nameservers
- success if the packet is “expected”
  - source IP addr: auth. nameserver
  - destination IP addr: resolver
  - source port: 53
  - destination port: fixed
  - query ID: guess

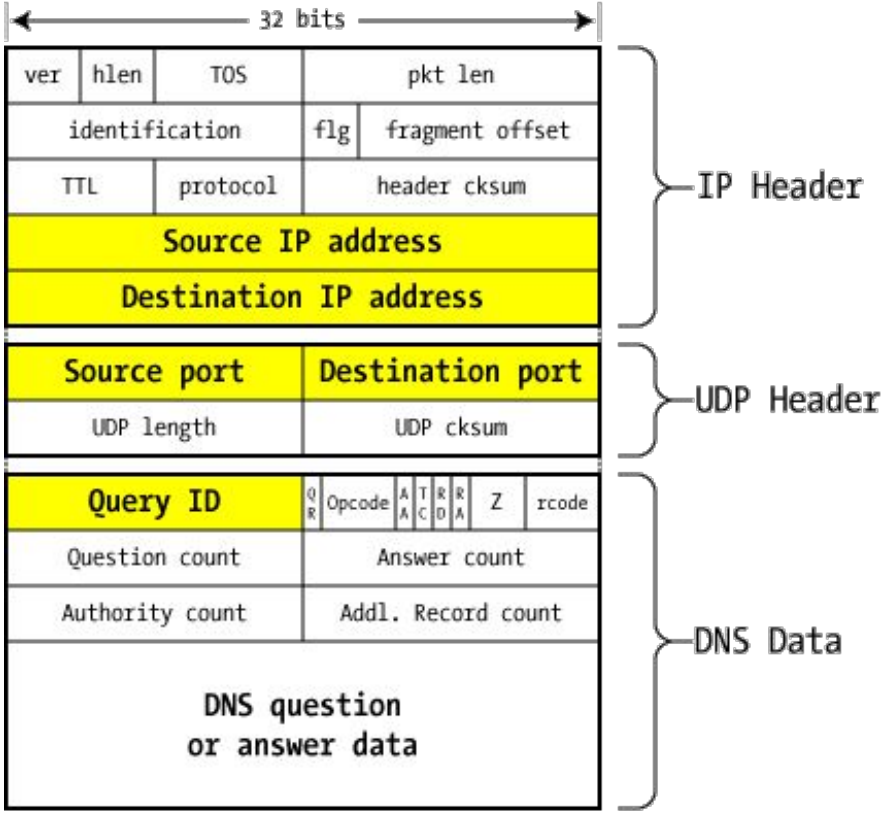


Figure 7

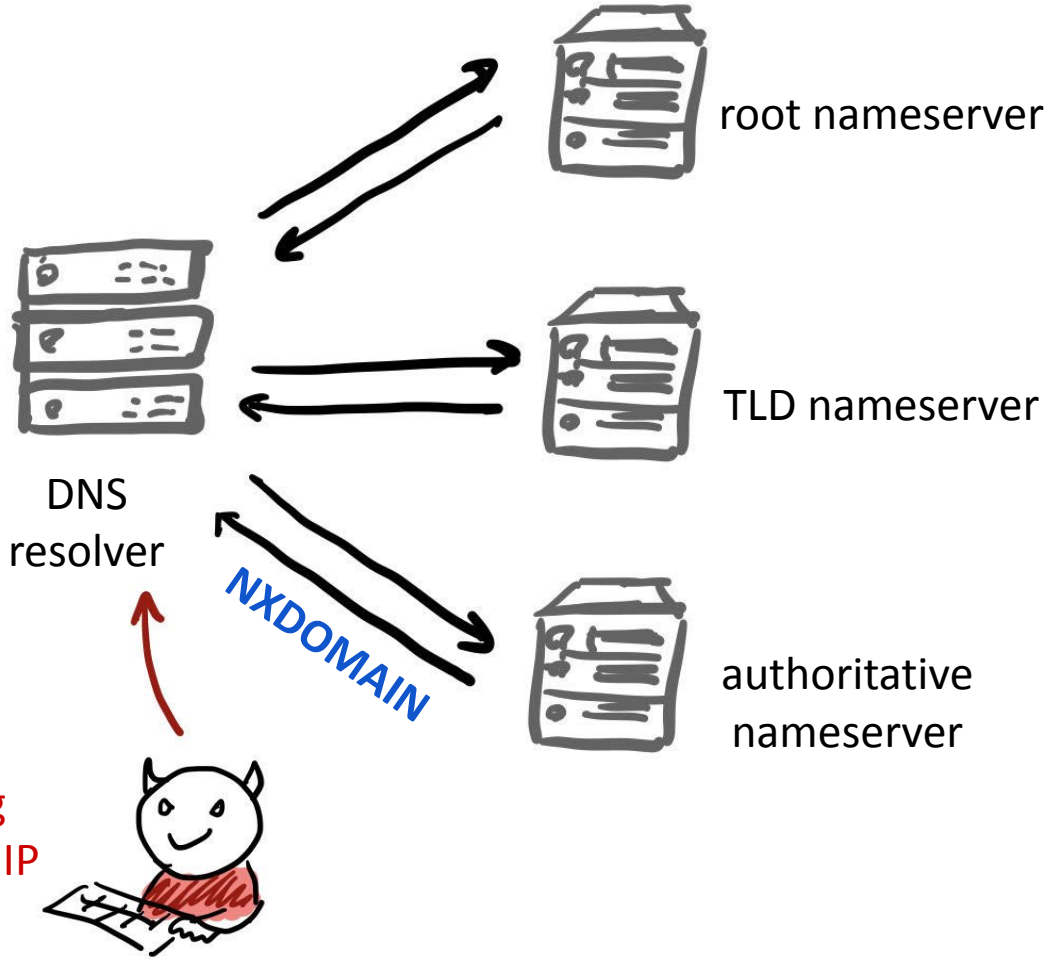
Limitation

- if fail once
  - the real response will arrive and be cached
  - need to wait before the attacker can make another attempt (TTL)
  - lose the game!

Kaminsky Attack

- perform the attack without waiting
    - ask the resolver to look up aaaaa.hitcon.org, bbbbb.hitcon.org, ccccc.hitcon.org … (a random host within hitcon.org domain)
    - each new query starts a new race
- RFC 1035: authority section contains RRs that point toward an authoritative name server
- what if we tell resolver that the nameserver for the hitcon.org is our machine ns.bad.org … ?

Kaminsky Attack



(1) IP address for
aaaaa.hitcon.org
bbbbb.hitcon.org
ccccc.hitcon.org
… ?

(2)
hitcon.org NS ns.bad.org
ns.bad.org A attacker's IP

Figure 8

Lab Environment

- victim’s machine
  - resolver
- DNS server
  - bind9 DNS server
- attacker’s machine
  - bind9 DNS server: return 1.2.3.4 for any query

Attack Overview

```
while (!success)
    send_query()          // send query packet to resolver
    send_fake_response() // send fake response packet to resolver
```

- step 1: queries the resolver for a random name in the hitcon.org domain, e.g. aaaaa.hitcon.org
- step 2: floods resolver with a stream of spoofed DNS responses, each trying a different query ID. In the replies, the attacker also provides an NS record, showing ns.bad.org as the nameserver for the hitcon.org domain; the attacker owns this nameserver which returns 1.2.3.4 for any query
- step 3: if fail, go back to step 1, use different hostname in the query

Construct the Spoofed DNS Response

- leverage the strength of both Python and C
  - use Scapy to create the DNS packet
  - load the packet into a C program

query ID (offset = 0x1C)

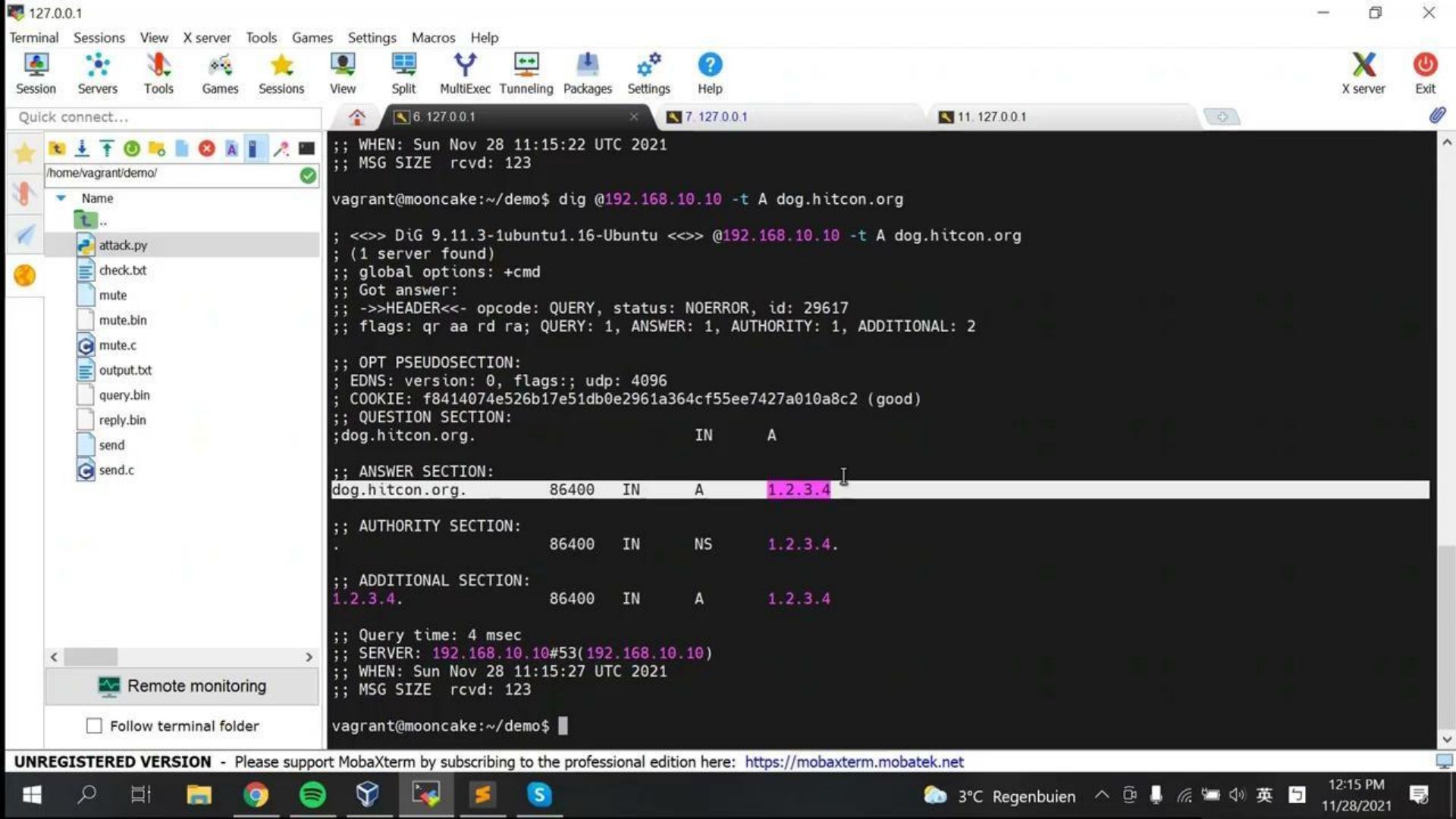
random hostname (offset = 0x29)

```
vagrant@mooncake:~/demo$ hexdump -C response.bin
00000000  45 00 00 9a 00 01 00 00  40 11 e4 d9 c0 a8 0a 1e  |E.......@.......|
00000010  c0 a8 0a 0a 00 35 25 37  00 86 6b b1 aa bb 85 00  |.....5%7..k.....|
00000020  00 01 00 01 00 01 00 01  05 61 61 61 61 61 06 68  |.........aaaaa.h|
00000030  69 74 63 6f 6e 03 6f 72  67 00 00 01 00 01 05 61  |itcon.org......a|
00000040  61 61 61 61 06 68 69 74  63 6f 6e 03 6f 72 67 00  |aaaa.hitcon.org.|
00000050  00 01 00 01 00 03 f4 80  00 04 01 02 03 04 06 68  |...............h|
00000060  69 74 63 6f 6e 03 6f 72  67 00 00 02 00 01 00 03  |itcon.org.......|
00000070  f4 80 00 0c 02 6e 73 03  62 61 64 03 6f 72 67 00  |.....ns.bad.org.|
00000080  02 6e 73 03 62 61 64 03  6f 72 67 00 00 01 00 01  |.ns.bad.org.....|
00000090  00 03 f4 80 00 04 c0 a8  0a 14                    |..........|
0000009a
```

Extend the Attack Window

- CCS'20: DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels
- Response rate limit (RRL)
  - authoritative nameserver has a RRL
  - “mute” the nameserver

Demo


127.0.0.1
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
6. 127.0.0.1
7. 127.0.0.1
11. 127.0.0.1
/home/vagrant/demo/
Name
..
attack.py
check.txt
mute
mute.bin
mute.c
output.txt
query.bin
reply.bin
send
send.c
Remote monitoring
Follow terminal folder
;; WHEN: Sun Nov 28 11:15:22 UTC 2021
;; MSG SIZE rcvd: 123
vagrant@mooncake:~/demo$ dig @192.168.10.10 -t A dog.hitcon.org
; <<>> DiG 9.11.3-1ubuntu1.16-Ubuntu <<>> @192.168.10.10 -t A dog.hitcon.org
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29617
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: f8414074e526b17e51db0e2961a364cf55ee7427a010a8c2 (good)
;; QUESTION SECTION:
;dog.hitcon.org. IN A
;; ANSWER SECTION:
dog.hitcon.org. 86400 IN A 1.2.3.4
;; AUTHORITY SECTION:
. 86400 IN NS 1.2.3.4.
;; ADDITIONAL SECTION:
1.2.3.4. 86400 IN A 1.2.3.4
;; Query time: 4 msec
;; SERVER: 192.168.10.10#53(192.168.10.10)
;; WHEN: Sun Nov 28 11:15:27 UTC 2021
;; MSG SIZE rcvd: 123
vagrant@mooncake:~/demo$
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
3°C Regenbuien
12:15 PM
11/28/2021

Mitigation

- source port randomization
  - most effective and widely deployed
  - randomness: 16-bit -> 32-bit
- Domain Name Security Extension (DNSSEC)
  - deployment rate (far from satisfactory)

In 2021, from Kaminsky attack to SAD DNS

- CCS'20: DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels
  - defeat the source port randomization via ICMP error messages
  - https://www.cs.ucr.edu/~zhiyunq/pub/ccs20_dns_poisoning.pdf
- CCS'21: DNS Cache Poisoning Attack: Resurrections with Side Channels
  - https://dl.acm.org/doi/pdf/10.1145/3460120.3486219

Reference

- DNS Security, Purdue University
  https://www.cs.purdue.edu/homes/ninghui/courses/526_Fall13/handouts/13_526_topic19.pdf
- Computer & Internet Security: A Hands-on Approach, Wenliang Du
- An Illustrated Guide to the Kaminsky DNS Vulnerability
  http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html