Poison Over Troubled Forwarders:

A Cache Poisoning Attack Targeting DNS Forwarding Devices

Xiaofeng Zheng, Chaoyi Lu, Jian Peng, Qiushi Yang, Dongjie Zhou, Baojun Liu, Keyu Man, Shuang Hao, Haixin Duan and Zhiyun Qian

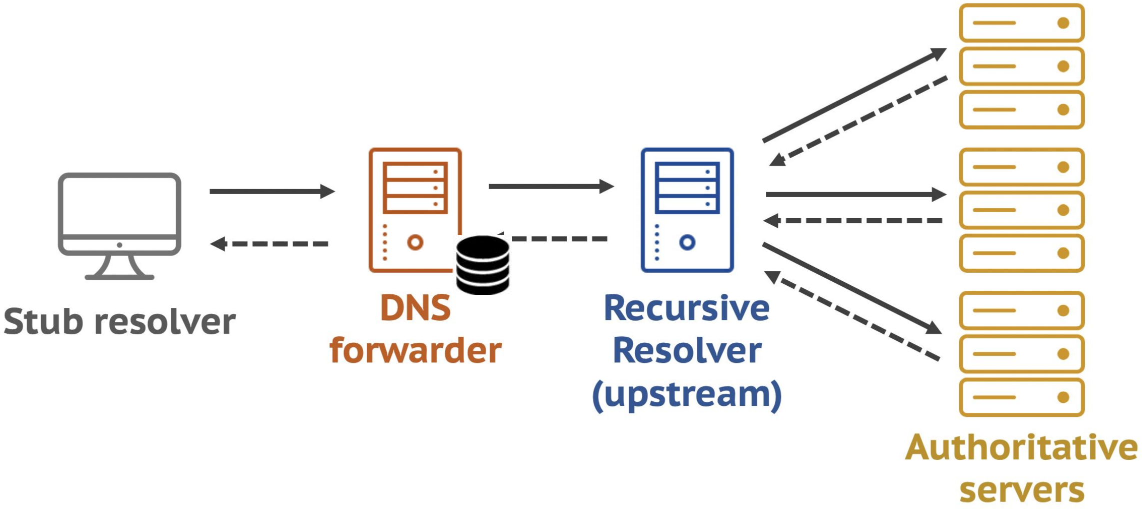
DNS Forwarder

- Devices standing in between stub and recursive resolvers

  E.g., home routers, open Wi-Fi networks

  Can have caching abilities

  Relies on the integrity of upstream resolvers



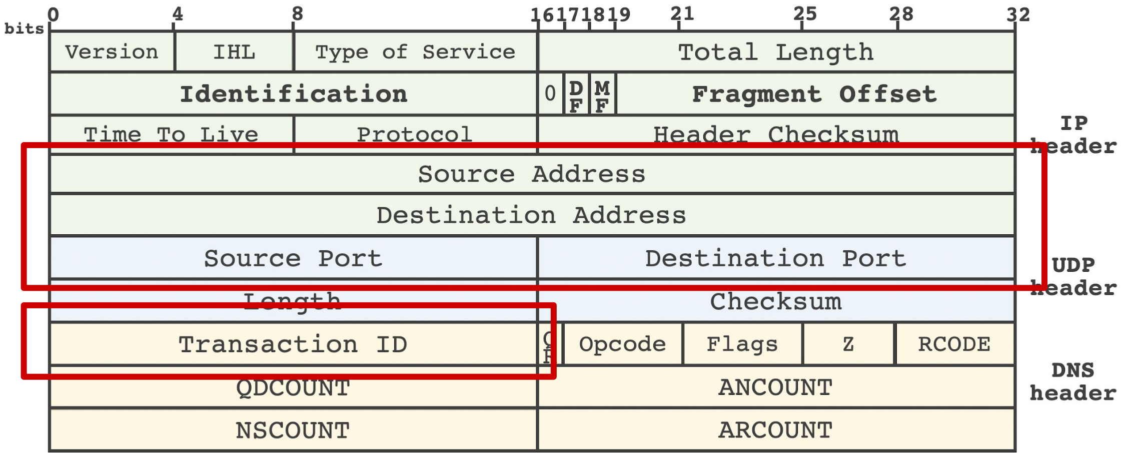
DNS Cache Poisoning Attacks

- Forging attacks targeting recursive resolvers

  Craft a DNS answer which matches the query’s metadata

  Example: Kaminsky Attack (2008)

  Mitigation: **increase randomness of DNS packet**



Version | IHL | Type of Service | Total Length

Identification | 0 | DF | MF | Fragment Offset

Time To Live | Protocol | Header Checksum

Source Address

Destination Address

Source Port | Destination Port

Length | Checksum

Transaction ID | QR | Opcode | Flags | Z | RCODE

QDCOUNT | ANCOUNT

NSCOUNT | ARCOUNT

RFC 5452:

DNS resolver implementations should use ***randomized*** *ephemeral port numbers and DNS transaction IDs*

Threat Model: Overview

- Defragmentation attacks targeting DNS forwarders

  Reliably forces DNS response fragmentation

  Targets **arbitrary victim domain names**

Threat Model: Overview

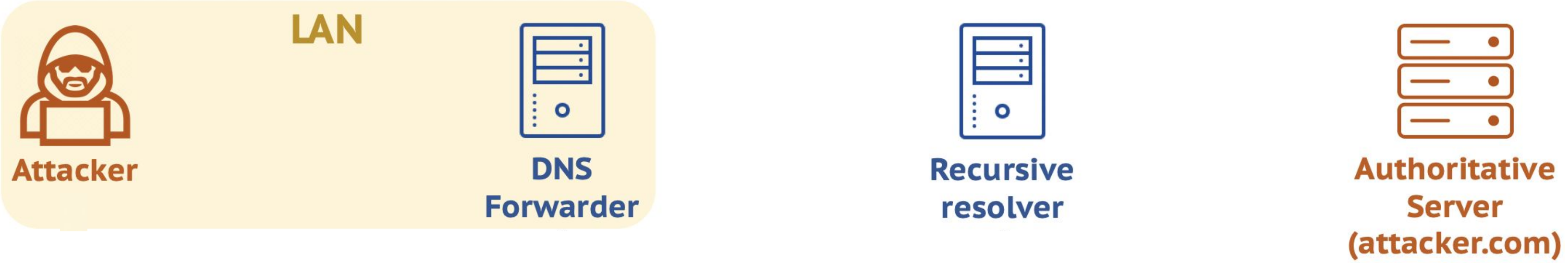
- Defragmentation attacks targeting DNS forwarders

  Reliably forces DNS response fragmentation

  Targets **arbitrary victim domain names**

1. Attacker & DNS forwarder locate in the same LAN
(e.g., in open Wi-Fi networks)

2. Use attacker’s own domain name and authoritative server



Insight on Forwarder Roles

- Defragmentation attacks targeting DNS forwarders

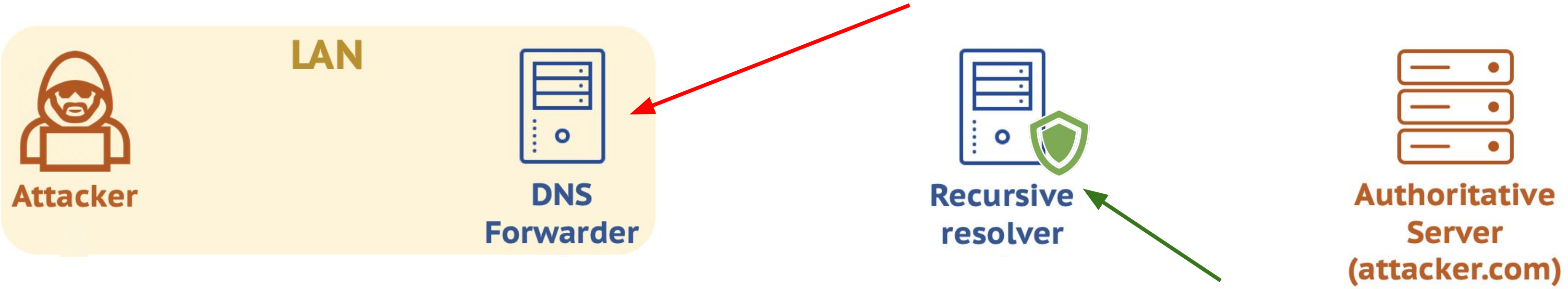
  Reliably forces DNS response fragmentation

  Targets **arbitrary victim domain names**

1. Attacker & DNS forwarder locate in the same LAN
(e.g., in open Wi-Fi networks)

Relies on recursive resolvers
Target of cache poisoning

2. Use attacker’s own domain name and authoritative server

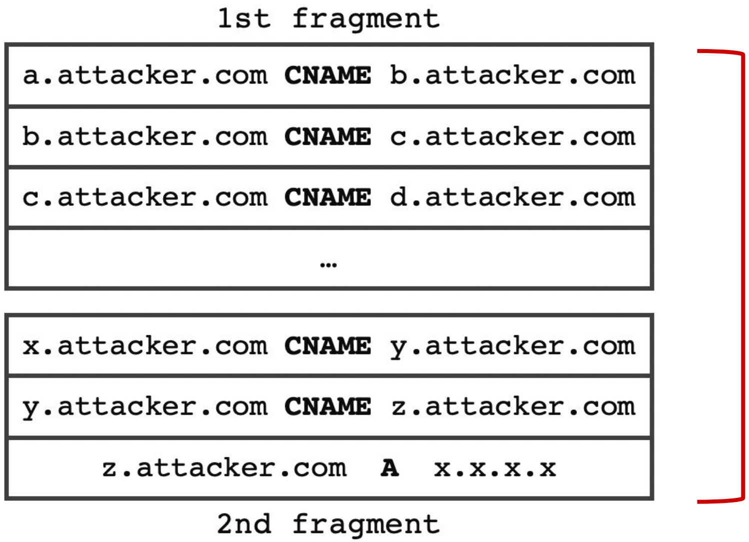


Security checks
(e.g., DNSSEC)

Attacker's Oversized DNS Response

- CNAME chain

Use dummy **CNAME records** to enlarge attacker's DNS response



1st fragment

a.attacker.com **CNAME** b.attacker.com

b.attacker.com **CNAME** c.attacker.com

c.attacker.com **CNAME** d.attacker.com

…

x.attacker.com **CNAME** y.attacker.com

y.attacker.com **CNAME** z.attacker.com

z.attacker.com **A** x.x.x.x

2nd fragment

> 1,500 Bytes (Ethernet MTU)

Always produce fragments

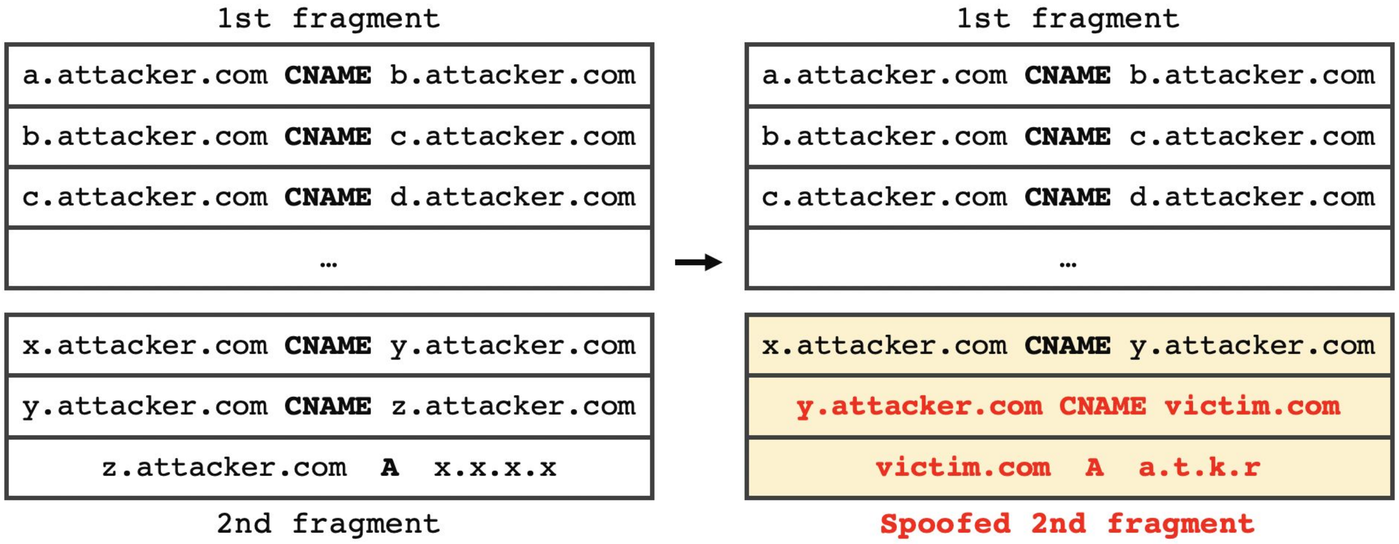
Attacker's Oversized DNS Response

- CNAME chain

  Use dummy **CNAME records** to enlarge attacker's DNS response

  Use CNAME to **point attacker's domain to any victim**

What the recursive resolver sees



1st fragment

```
a.attacker.com CNAME b.attacker.com
b.attacker.com CNAME c.attacker.com
c.attacker.com CNAME d.attacker.com
…
```

```
x.attacker.com CNAME y.attacker.com
y.attacker.com CNAME z.attacker.com
z.attacker.com  A  x.x.x.x
```

2nd fragment

→

1st fragment

```
a.attacker.com CNAME b.attacker.com
b.attacker.com CNAME c.attacker.com
c.attacker.com CNAME d.attacker.com
…
```

```
x.attacker.com CNAME y.attacker.com
y.attacker.com CNAME victim.com
victim.com  A  a.t.k.r
```

Spoofed 2nd fragment

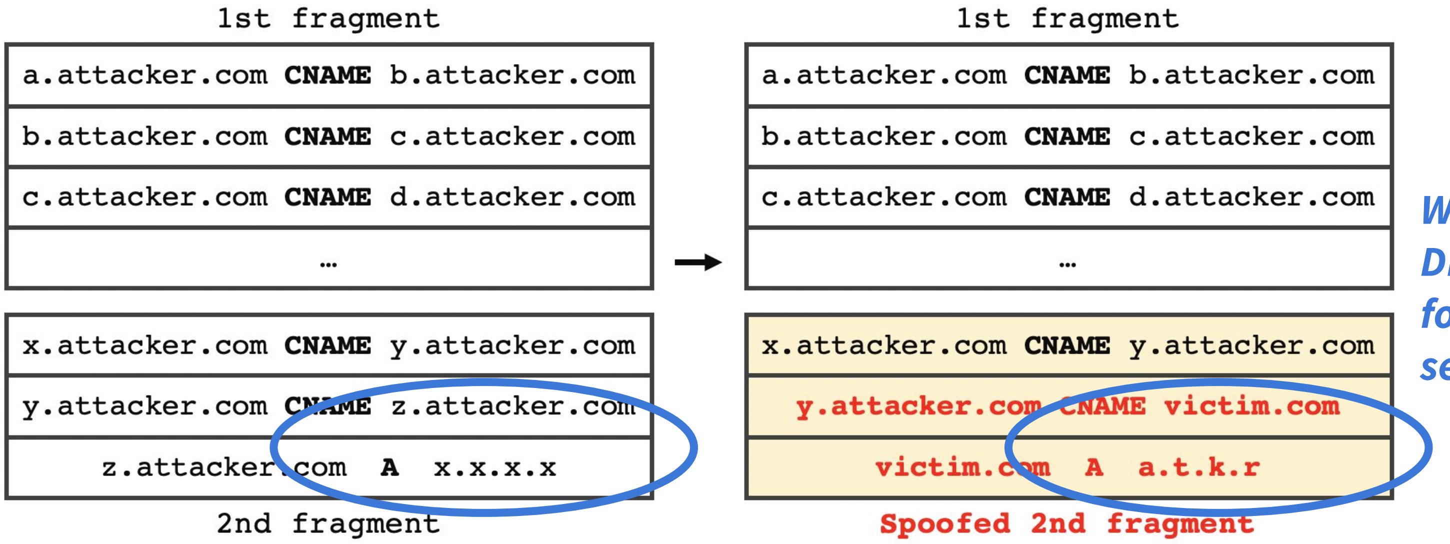
What the DNS forwarder sees

Attacker’s Oversized DNS Response

- CNAME chain

  Use dummy **CNAME records** to enlarge attacker’s DNS response

  Use CNAME to **point attacker’s domain to any victim**



What the recursive resolver sees

```
1st fragment
a.attacker.com CNAME b.attacker.com
b.attacker.com CNAME c.attacker.com
c.attacker.com CNAME d.attacker.com
…
x.attacker.com CNAME y.attacker.com
y.attacker.com CNAME z.attacker.com
z.attacker.com  A  x.x.x.x
2nd fragment
```

→

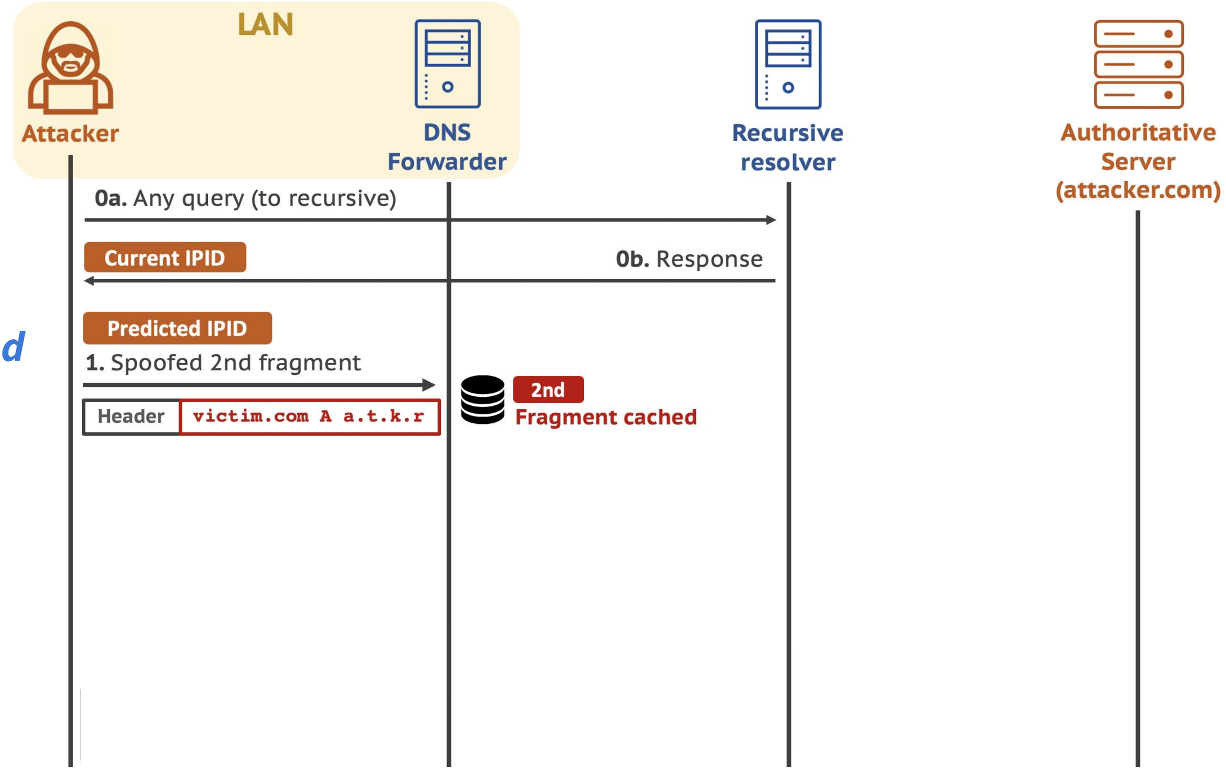
What the DNS forwarder sees

```
1st fragment
a.attacker.com CNAME b.attacker.com
b.attacker.com CNAME c.attacker.com
c.attacker.com CNAME d.attacker.com
…
x.attacker.com CNAME y.attacker.com
y.attacker.com CNAME victim.com
victim.com  A  a.t.k.r
Spoofed 2nd fragment
```

Flow of Defragmentation Attack

- Defragmentation attacks targeting DNS forwarders

1. Craft spoofed 2nd fragment

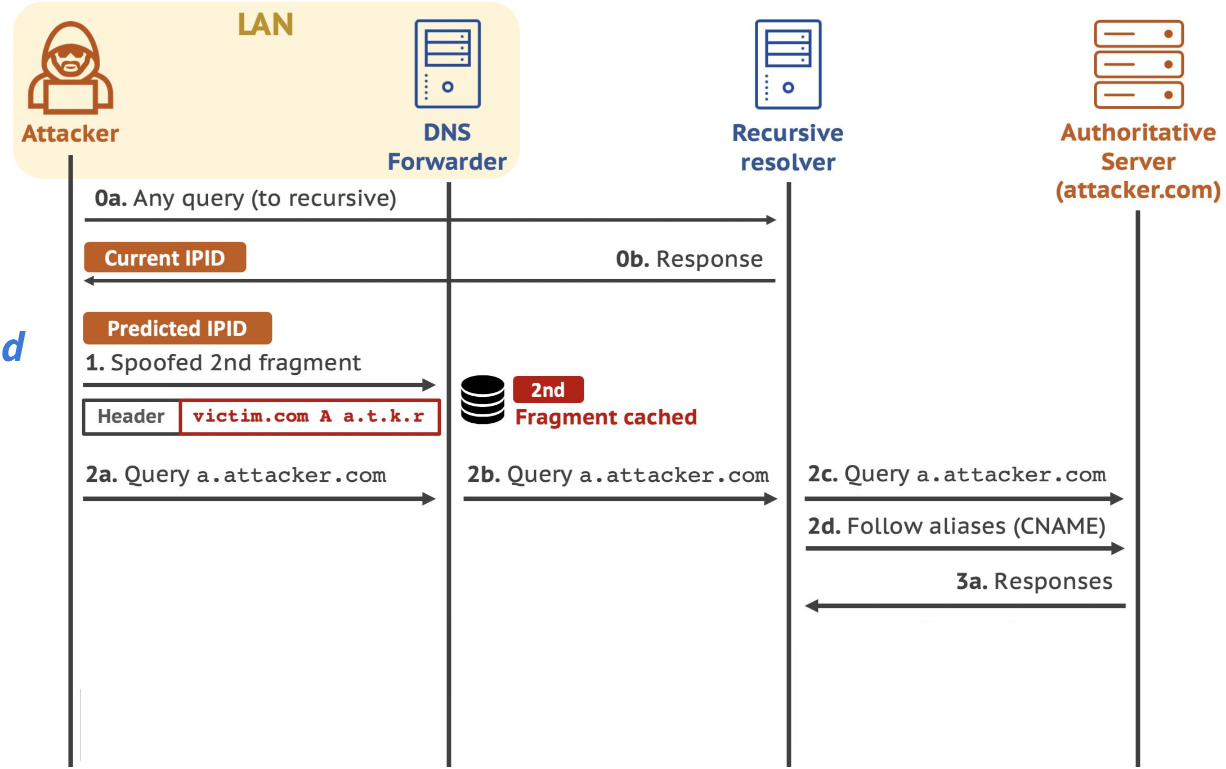


Flow of Defragmentation Attack

- Defragmentation attacks targeting DNS forwarders

1. Craft spoofed 2nd fragment

2. Issue a DNS query

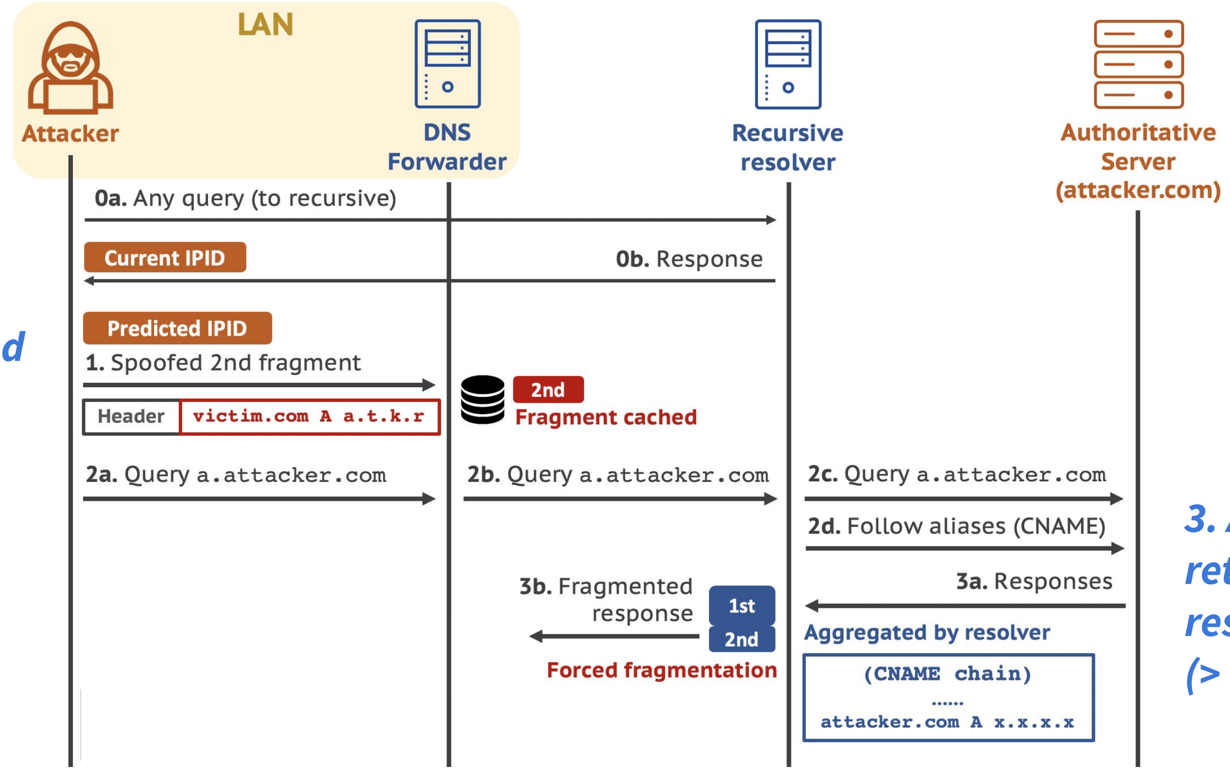


Flow of Defragmentation Attack

- Defragmentation attacks targeting DNS forwarders

1. Craft spoofed 2nd fragment

2. Issue a DNS query


LAN
Attacker
DNS Forwarder
Recursive resolver
Authoritative Server (attacker.com)
0a. Any query (to recursive)
Current IPID
0b. Response
Predicted IPID
1. Spoofed 2nd fragment
Header
victim.com A a.t.k.r
2nd
Fragment cached
2a. Query a.attacker.com
2b. Query a.attacker.com
2c. Query a.attacker.com
2d. Follow aliases (CNAME)
3a. Responses
3b. Fragmented response
1st
2nd
Forced fragmentation
Aggregated by resolver
(CNAME chain)
......
attacker.com A x.x.x.x

3. Authoritative returns oversized response (> Ethernet MTU)

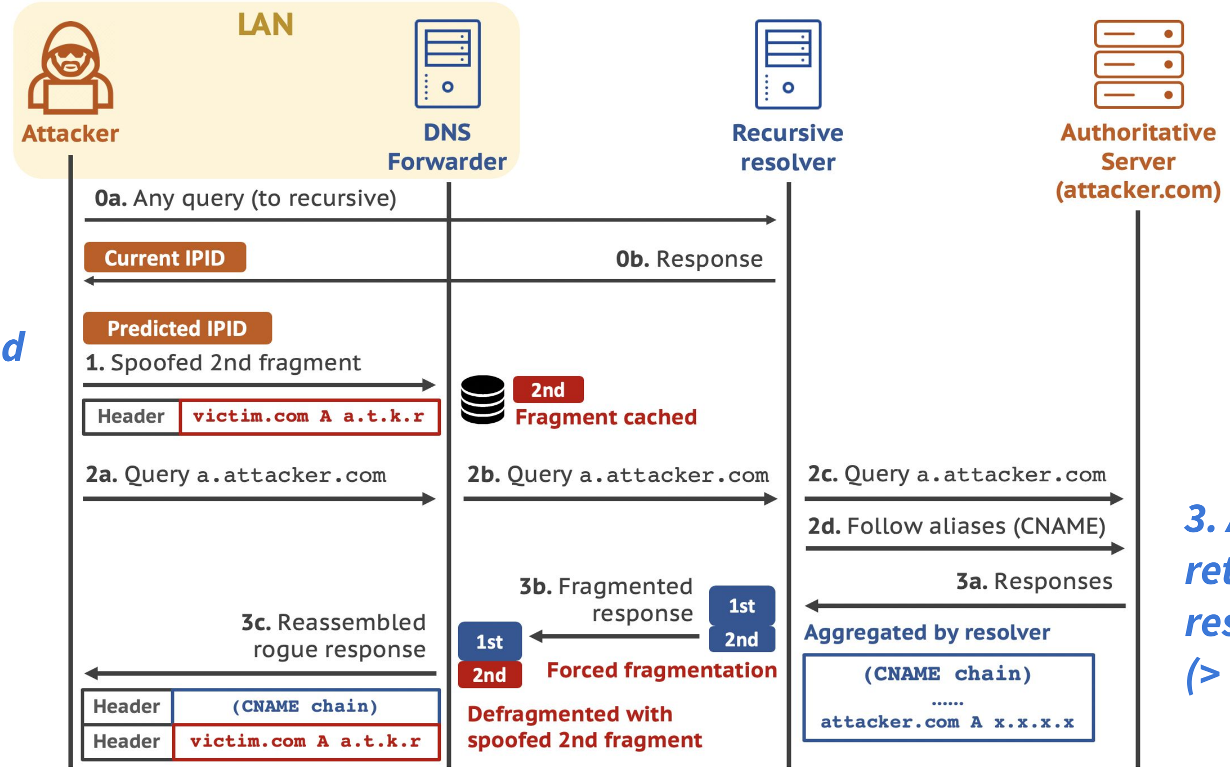
Flow of Defragmentation Attack

- Defragmentation attacks targeting DNS forwarders

1. Craft spoofed 2nd fragment

2. Issue a DNS query

4. Defragment by forwarder



3. Authoritative returns oversized response (> Ethernet MTU)

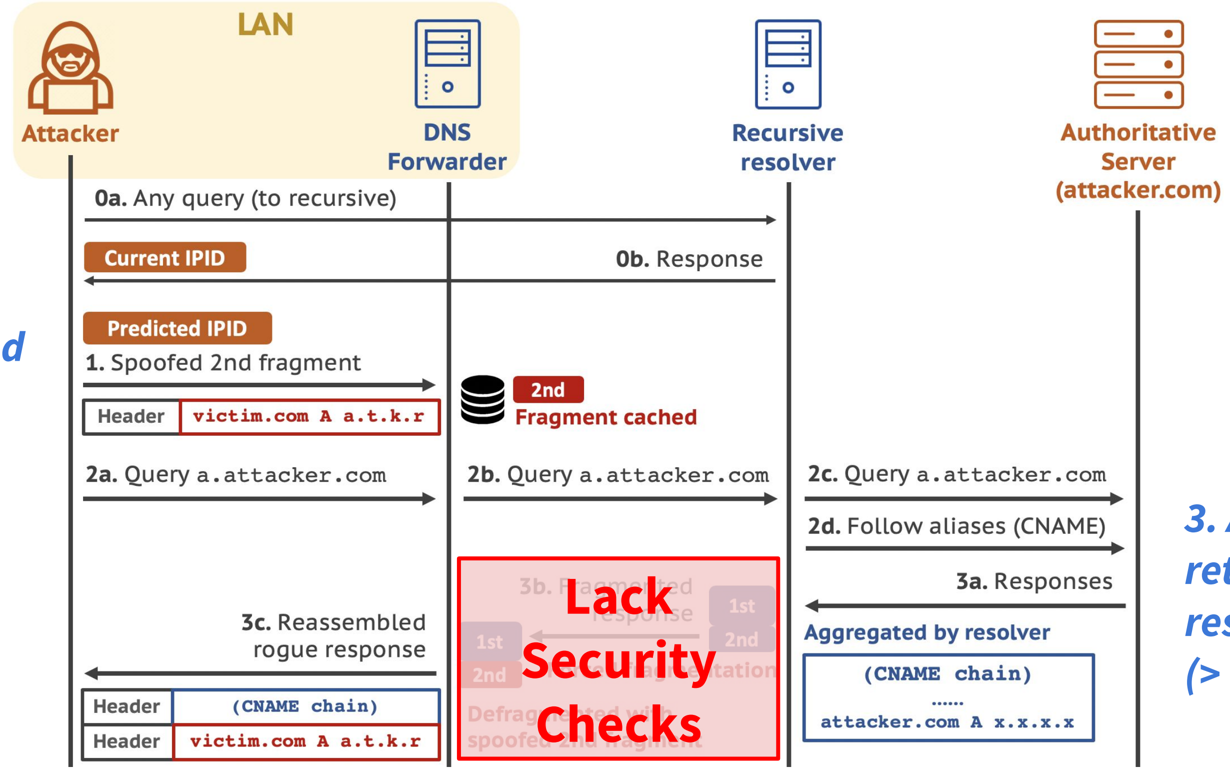
Flow of Defragmentation Attack

- Defragmentation attacks targeting DNS forwarders

1. Craft spoofed 2nd fragment

2. Issue a DNS query

4. Defragment by forwarder



LAN

Attacker

DNS Forwarder

Recursive resolver

Authoritative Server (attacker.com)

0a. Any query (to recursive)

Current IPID

0b. Response

Predicted IPID

1. Spoofed 2nd fragment

Header | `victim.com A a.t.k.r`

2nd Fragment cached

2a. Query `a.attacker.com`

2b. Query `a.attacker.com`

2c. Query `a.attacker.com`

2d. Follow aliases (CNAME)

3a. Responses

Aggregated by resolver

`(CNAME chain)`

`......`

`attacker.com A x.x.x.x`

3b. Fragmented response

1st

2nd

1st

2nd

IP defragmentation

Defragmented with spoofed 2nd fragment

Lack Security Checks

3c. Reassembled rogue response

Header | `(CNAME chain)`

Header | `victim.com A a.t.k.r`

3. Authoritative returns oversized response (> Ethernet MTU)

Conditions of Successful Attacks

- DNS caching by record

  The tampered record can be cached separately

- EDNS(0) support

  Allows transfer of DNS messages larger than 512 Bytes

- No active truncation of DNS response

  Ensures that the entire oversized response is transfered

- No response verification

  DNS forwarders rely on upstream resolvers

Vulnerable DNS Software

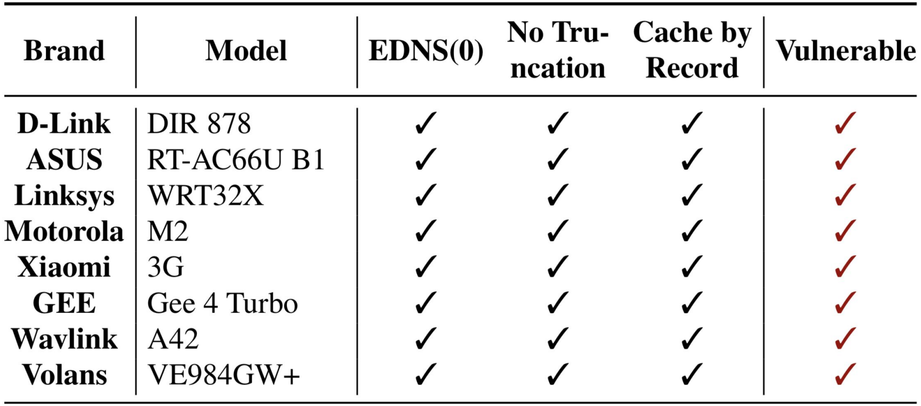
- Home routers

  16 models are tested (by real attacks in controlled environment)

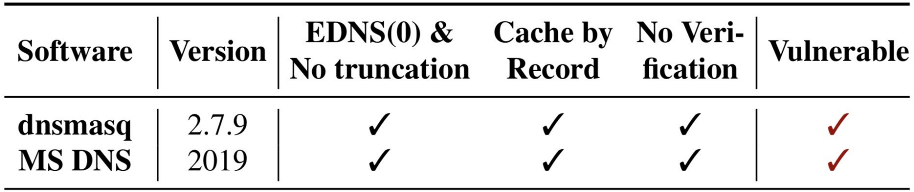
  8 models are vulnerable

- DNS software

  2 kinds of popular DNS software are vulnerable



Brand	Model	EDNS(0)	No Truncation	Cache by Record	Vulnerable
D-Link	DIR 878	✓	✓	✓	✓
ASUS	RT-AC66U B1	✓	✓	✓	✓
Linksys	WRT32X	✓	✓	✓	✓
Motorola	M2	✓	✓	✓	✓
Xiaomi	3G	✓	✓	✓	✓
GEE	Gee 4 Turbo	✓	✓	✓	✓
Wavlink	A42	✓	✓	✓	✓
Volans	VE984GW+	✓	✓	✓	✓



Software	Version	EDNS(0) & No truncation	Cache by Record	No Verification	Vulnerable
dnsmasq	2.7.9	✓	✓	✓	✓
MS DNS	2019	✓	✓	✓	✓

Vulnerable DNS Software

- Home routers

  16 models are tested (by real attacks in controlled environment)

  8 models are vulnerable

- DNS software

  2 kinds of popular DNS software are vulnerable

- Responsible Disclosure

  ASUS and D-Link release firmware patches

  Linksys accepts issue via BugCrowd

Measuring Clients Potentially Under Risk

- Collect vantage points

  Implement measurement code in a network diagnosis tool

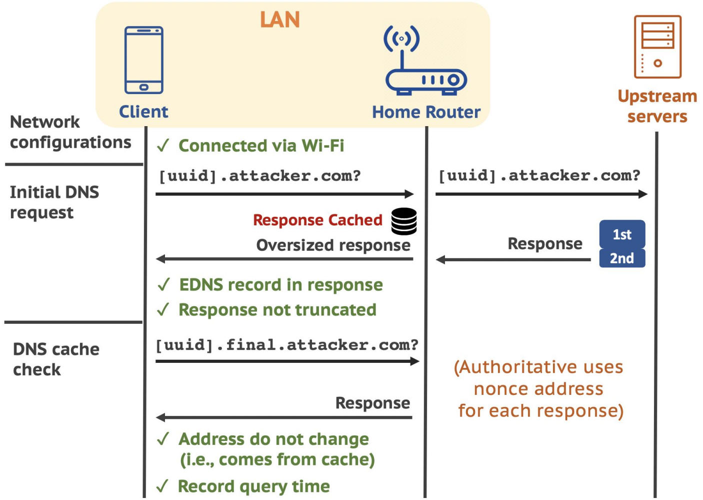
  20K clients, mostly located in China

- Check the forwarder conditions

  Ethical considerations: no real attack

  40% do not support EDNS(0) yet

  Estimated vulnerable clients: 6.6%



LAN

Client

Home Router

Upstream servers

Network configurations

✓ Connected via Wi-Fi

Initial DNS request

`[uuid].attacker.com?`

`[uuid].attacker.com?`

Response Cached

Oversized response

Response

1st

2nd

✓ EDNS record in response

✓ Response not truncated

DNS cache check

`[uuid].final.attacker.com?`

Response

(Authoritative uses nonce address for each response)

✓ Address do not change (i.e., comes from cache)

✓ Record query time

Discussion

- Mitigation for DNS forwarders
    - Perform response verification (e.g., DNSSEC)
    - **DNS caching by response (short-term solution)**
- Lack clear guidelines of DNS forwarders
    - What role should they play?
    - What features should be supported?

- An attack targeting DNS forwarders
- Affects forwarder implementations extensively
- Call for more attention on DNS forwarder security

Any Questions?

zxf19@mails.tsinghua.edu.cn