数字寰宇
SHUZIHUANYU

MaginotDNS 攻击：跨越域名解析器的缓存防御“护城河”

数字寰宇大家讲堂公开课

分享人：李想 清华大学

2023-09-13

Xiang Li (李想)

- **5th-year Ph.D. Candidate**
  - ❑ Tsinghua University (NISL Lab), UCI (visiting scholar)
  - ❑ Advisor(s): Prof. Qi Li and Haixin Duan
- **Research Area and Publication**
  - ❑ Network scanning, IPv6 security, DNS security, vulnerability discovery, and fuzzing
  - ❑ **Publications in total (12):** S&P ('24), NDSS ('23, '24), Security ('23a, '23b, '24), CCS ('23a, '23b), DSN ('21), VehicleSec ('23), SIGMETRICS ('23), IMC ('23)
  - ❑ **Publications as the 1st author (5):** S&P ('24), NDSS ('23), Security ('23), CCS ('23), DSN ('21)
  - ❑ **Publications as the corresponding author (1):** USENIX Security ('24)
  - ❑ **Industry conferences:** IDS ('21, '22), DNS OARC (39, 40, 41), Black Hat (AS '23, US '23)

Xiang Li (李想)

➢ Prize (Part)

- ❑ Tsinghua Outstanding 2nd Scholarship - 2022
- ❑ Outstanding Undergraduate - 2019
- ❑ Nankai Gongneng 1st Scholarship - 2018
- ❑ Cyber Security Scholarship of China Internet Development Foundation - 2018
- ❑ China National Scholarship - 2016, 2017

➢ Competition (Part)

- ❑ 1st/3rd/3rd Prize in IPv6 Technology Application Innovation Competition – 2022/2023
- ❑ 2nd Prize in National College Student Information Security Contest - 2018
- ❑ 3rd Prize in National Cryptography Contest - 2017

Xiang Li (李想)

- **CNVD/CNNVD/CVE**
  - **Total: 109/5/75**
  - **Bounty: US$11,600**
  - ResolverFuzz Vulnerability (2023): n/n/15
  - TuDoor Vulnerability (2023): n/n/32
  - TsuKing DoS Vulnerability (2023): n/n/3
  - Phoenix Domain Vulnerability (2022): n/n/9
  - MaginotDNS Cache Poisoning Vulnerability (2022): n/n/3
  - IPv6 Routing Loop Vulnerability (2021): 109/5/22

MaginotDNS 攻击：跨越域名解析器的缓存防御“护城河”

The Maginot Line: Attacking the Boundary of DNS Caching Protection

[Published at USENIX Security ’23]

Presenter: **Xiang Li** Tsinghua University

Sept. 2023

UCI Samueli School of Engineering
University of California, Irvine
奇安信 QI-ANXIN

Attack Impact

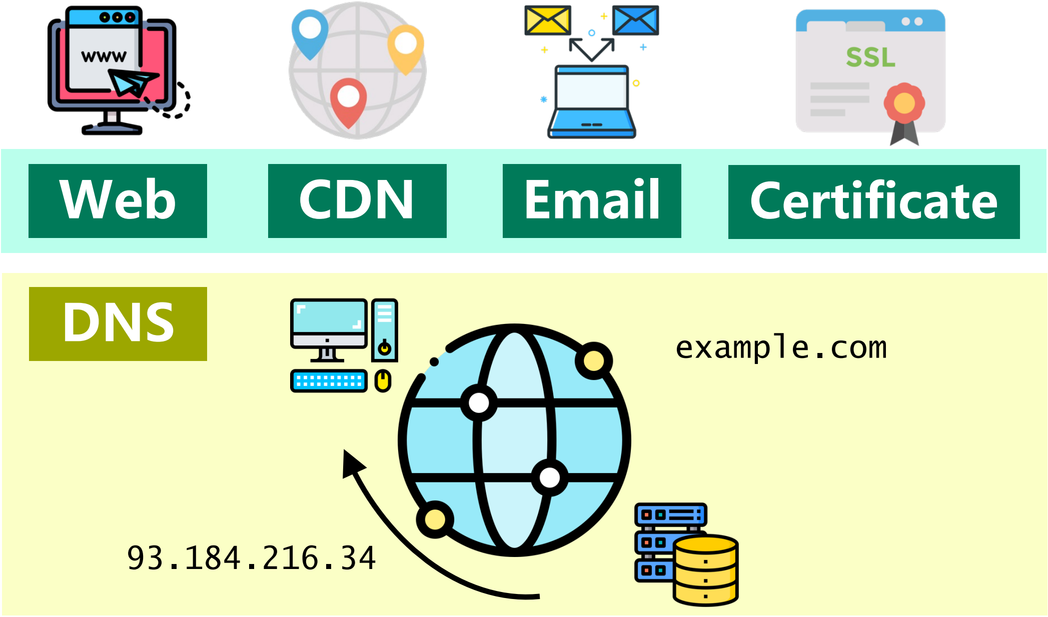
Our MaginotDNS attack could poison a whole TLD, e.g., .com and .net, at a time.

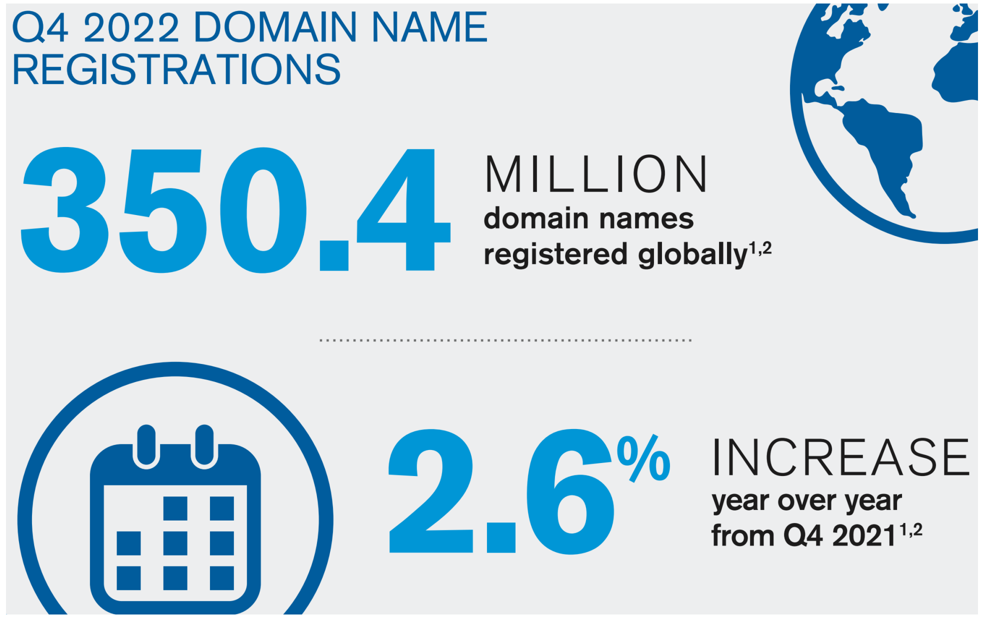
Thus, all domains under that TLD can be hijacked.

Domain Name System (DNS)

➢ DNS Overview

- ❑ Translating domain names to IP addresses
- ❑ Entry point of many Internet activities
- ❑ Domain names are widely registered





verisign.com/dnib

Domain Name System (DNS)

➢ Hierarchical Name Space

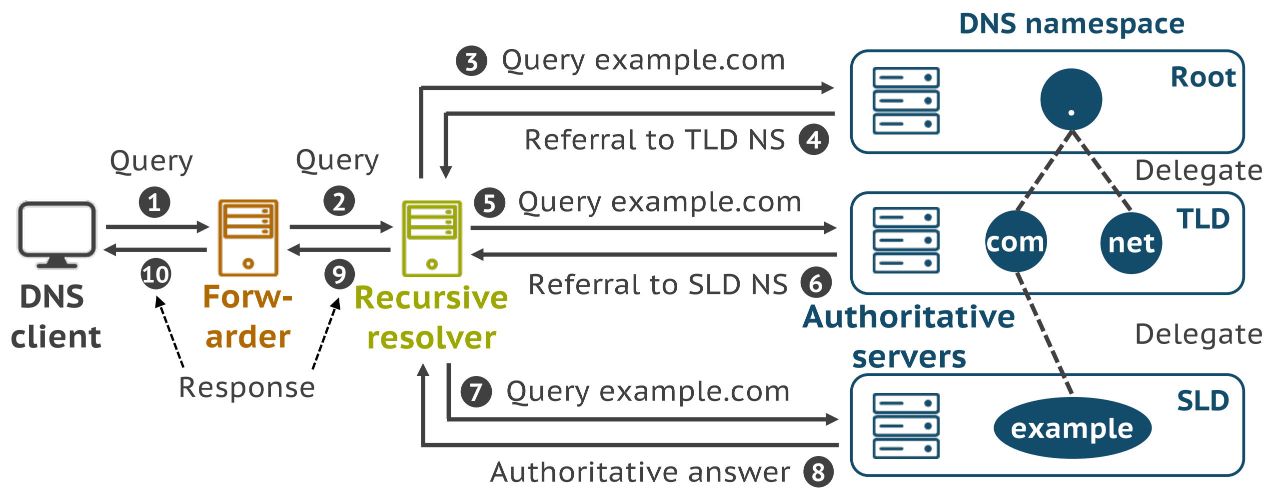
- ❑ Authoritative zones: root, TLD, SLD → DNS records
- ❑ Domain delegation → Domain registration

➢ Multiple Resolver Roles

- ❑ Client, forwarder, recursive, authoritative
- ❑ Caching

➢ Iterative Resolution Process

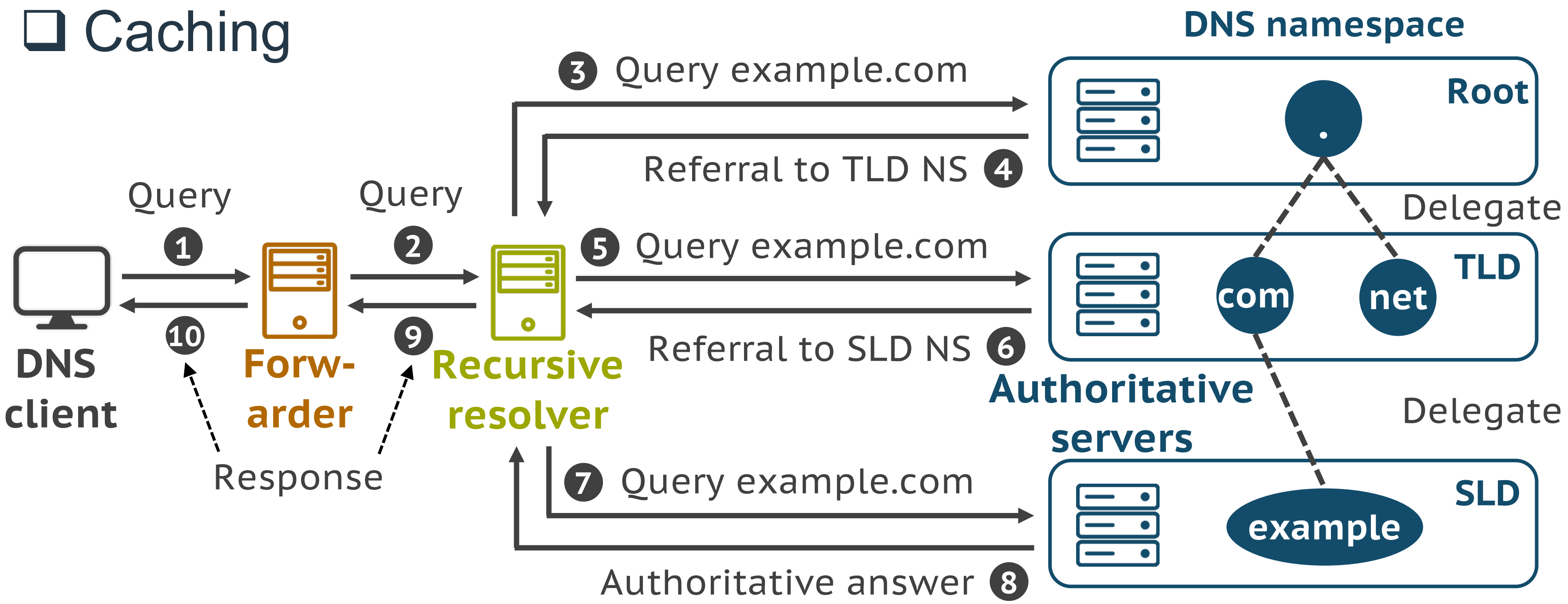
- ❑ Client-server style

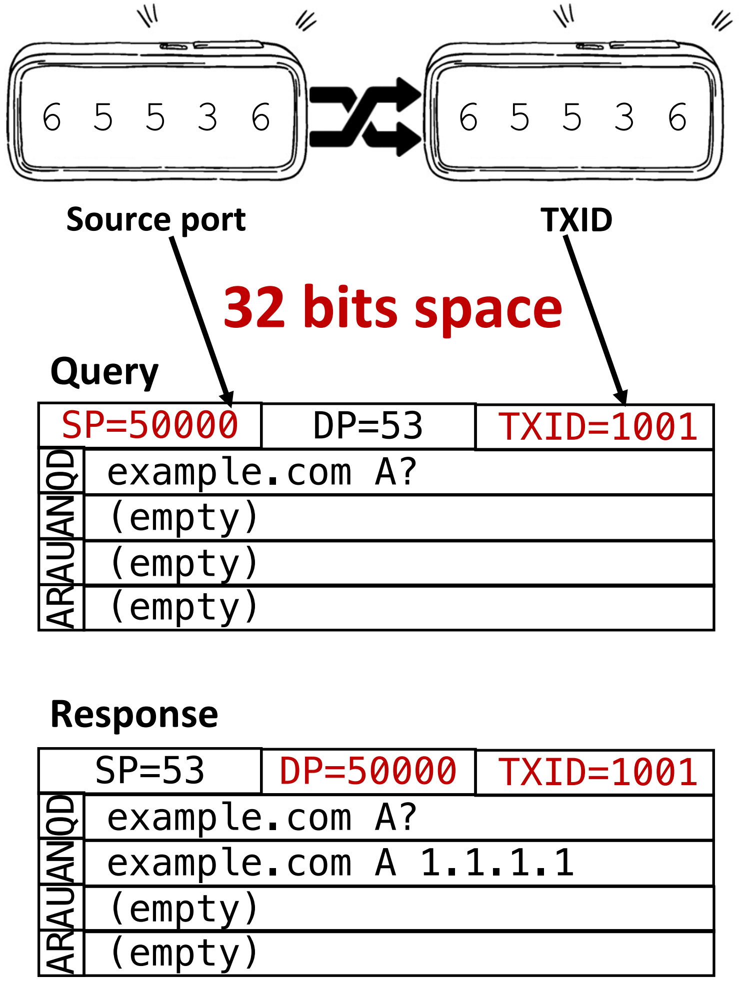


Domain Name System (DNS)

➢ DNS Resolution Process

- ❑ Primarily over UDP
- ❑ Iterative and recursive
- ❑ Caching





Takeaway

Since DNS is the cornerstone of the Internet, enabling multiple critical services and applications,

Attackers have long been trying to manipulate its response for hijacking via **cache poisoning attacks**.

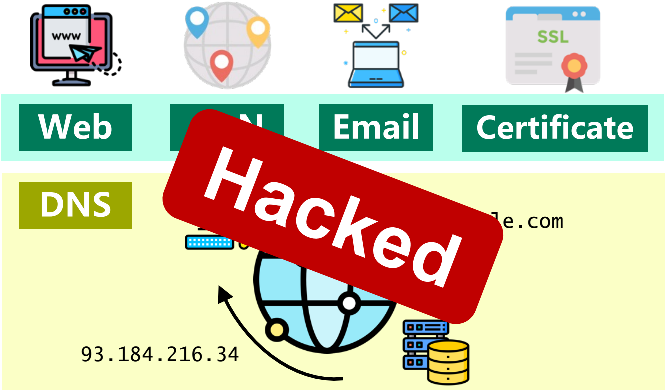
Question

What is DNS cache poisoning?

Since DNS is primarily over UDP, attackers want to **inject forged answers into resolvers' cache**.

DNS Cache Poisoning

- **Target**
  - ❑ Injecting forged answers into resolvers' cache
- **Taxonomy**
  - ❑ On-path, off-path
- **Technique**
  - ❑ Cat-and-mouse game



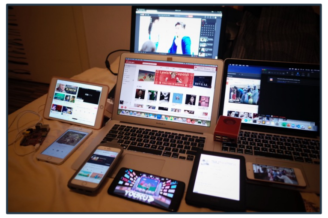
1997 Birthday Attack

2002 Kashpureff Attack

2008 Kaminsky Attack

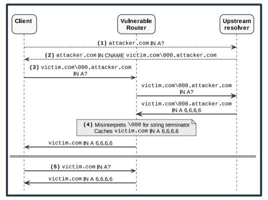
2013 Fragmentation Attack

2020 Attack on Forwarders

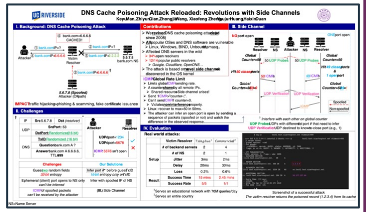


2020 SADDNS Attack

2021 Attack via Escaped Chars



2021 SADDNS v2 Attack

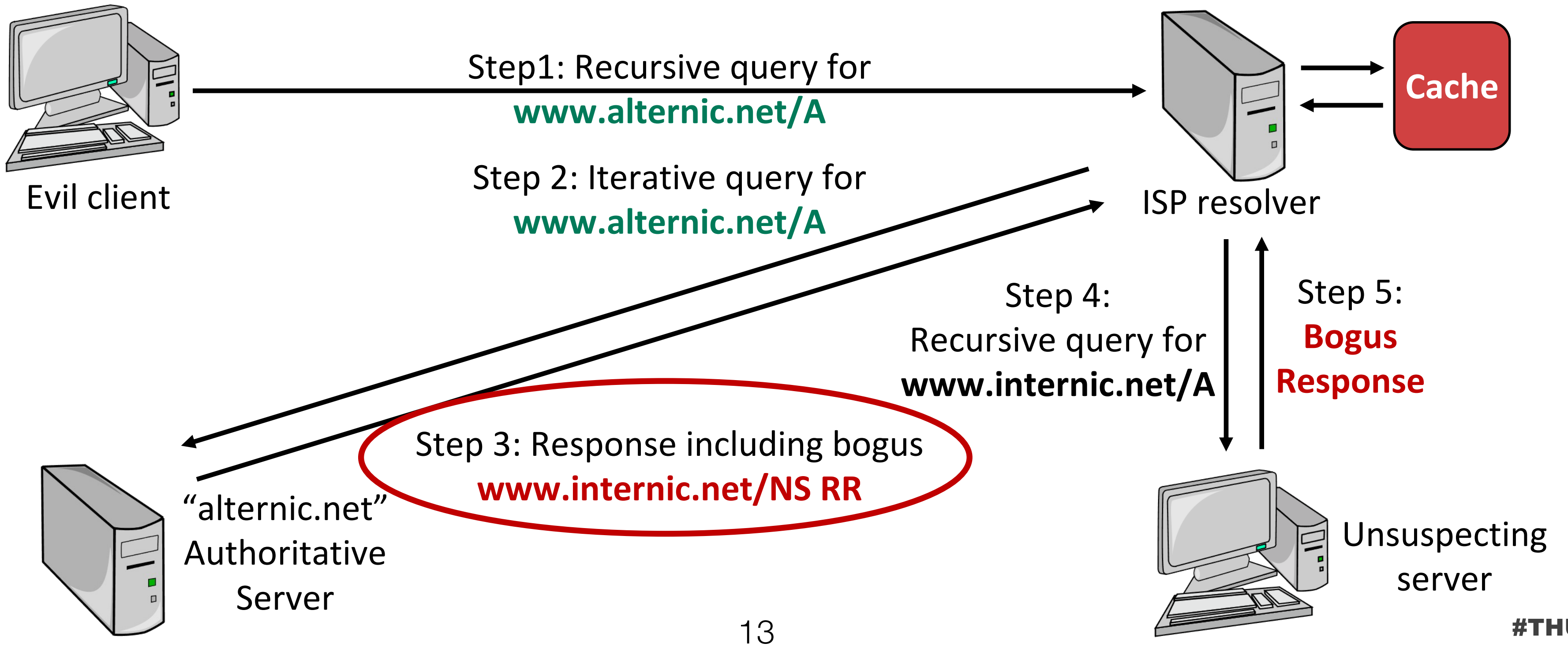


2022 MaginotDNS Attack

2023 Attack via Escaped Chars v2

DNS Cache Poisoning

- **Kashpureff Attack (on-path, 1997)**
  - ❑ Method: returning forged responses from the authoritative
  - ❑ Result: resolver accepting all records in the response
  - ❑ Cause: lacking data verification (**bailiwick rules**)



DNS Bailiwick Rules

- **Mitigating the Kashpureff Attack**
  - The credibility checking when storing cache entries
  - Checking for “**in bailiwick**” in response data: **answer records must be from the same domain as the requested name**

```
$ dig example.com

;; ANSWER SECTION:
example.com. 86400 IN A 93.184.216.34

;; AUTHORITY SECTION:
mybank.com. 86400 IN NS ns.mybank.com.

;; ADDITIONAL SECTION:
ns.mybank.com. 86400 IN A 1.2.3.4
```

Bailiwick (pointing to: example.com)

In-bailiwick
Can be trusted (pointing to the ANSWER SECTION record)

Out-of-bailiwick
Should be removed (pointing to the AUTHORITY and ADDITIONAL SECTION records, which are struck through)

Takeaway

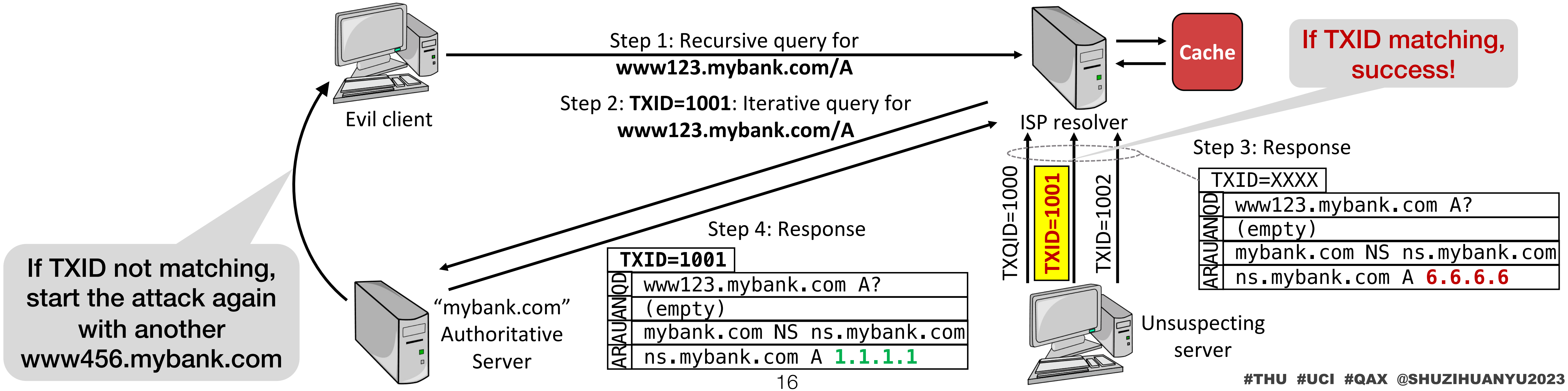
After the Kashpureff attack, bailiwick checking is integrated into the resolver's implementation,

DNS cache poisoning on recursives from the on-path seems **impossible** to conduct from 1997.

DNS Cache Poisoning

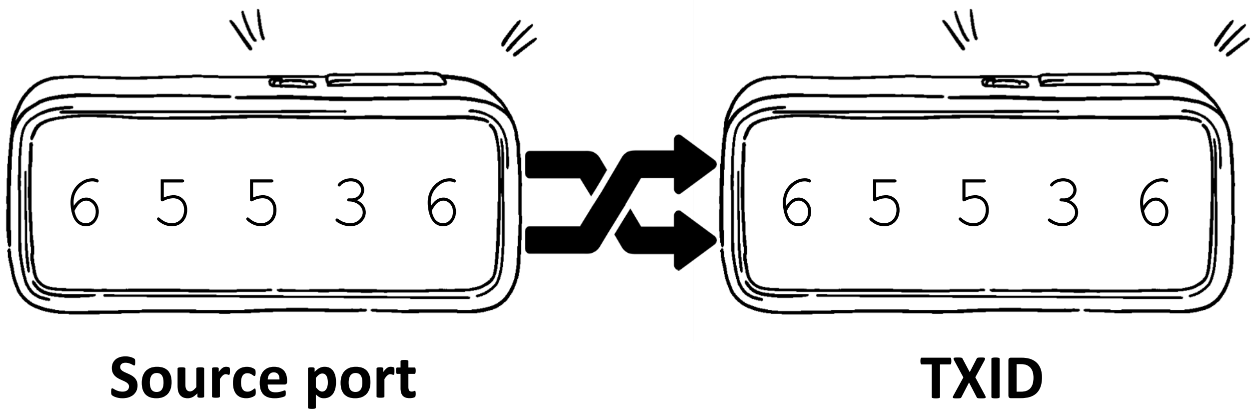
➢ Kaminsky Attack (Off-path, 2008)

- ❑ Method: injecting forged responses with the "birthday paradox"
- ❑ Result: resolver accepting glue records in the response
- ❑ Cause: lacking **source port randomization** (TXID only 16 bits)



DNS Source Port/TXID Randomization

- **Mitigating the Kaminsky Attack**
  - ❑ Increasing the query guessing entropy
  - ❑ 16-bit source port x 16-bit TXID = 32-bit space
  - ❑ **Hard to brute-force**



Takeaway

After the Kaminsky attack, source port randomization is integrated into the resolver's implementation,

DNS cache poisoning on resolvers from the off-path became **difficult** to conduct from 2008.

Question

26 years later, does bailiwick checking work as desired after fixing the Kashpureff attack?

No. **MaginotDNS** breaks this guarantee with a new powerful **cache poisoning vulnerability**.

MaginotDNS Attack

- **What is the MaginotDNS attack**
  - Proposed by our **NISL** lab, published at [USENIX Security ’23]
  - A new powerful DNS cache poisoning attack against **CDNS resolvers**
  - Can be launched from either **on-path** or **off-path**
  - Can poison **arbitrary domains** including **TLDs**, such as .com and .net
- **Name**
  - Exploiting **vulnerabilities** of bailiwick checking to bypass itself
  - Working like breaking the **Maginot Line** → **MaginotDNS**

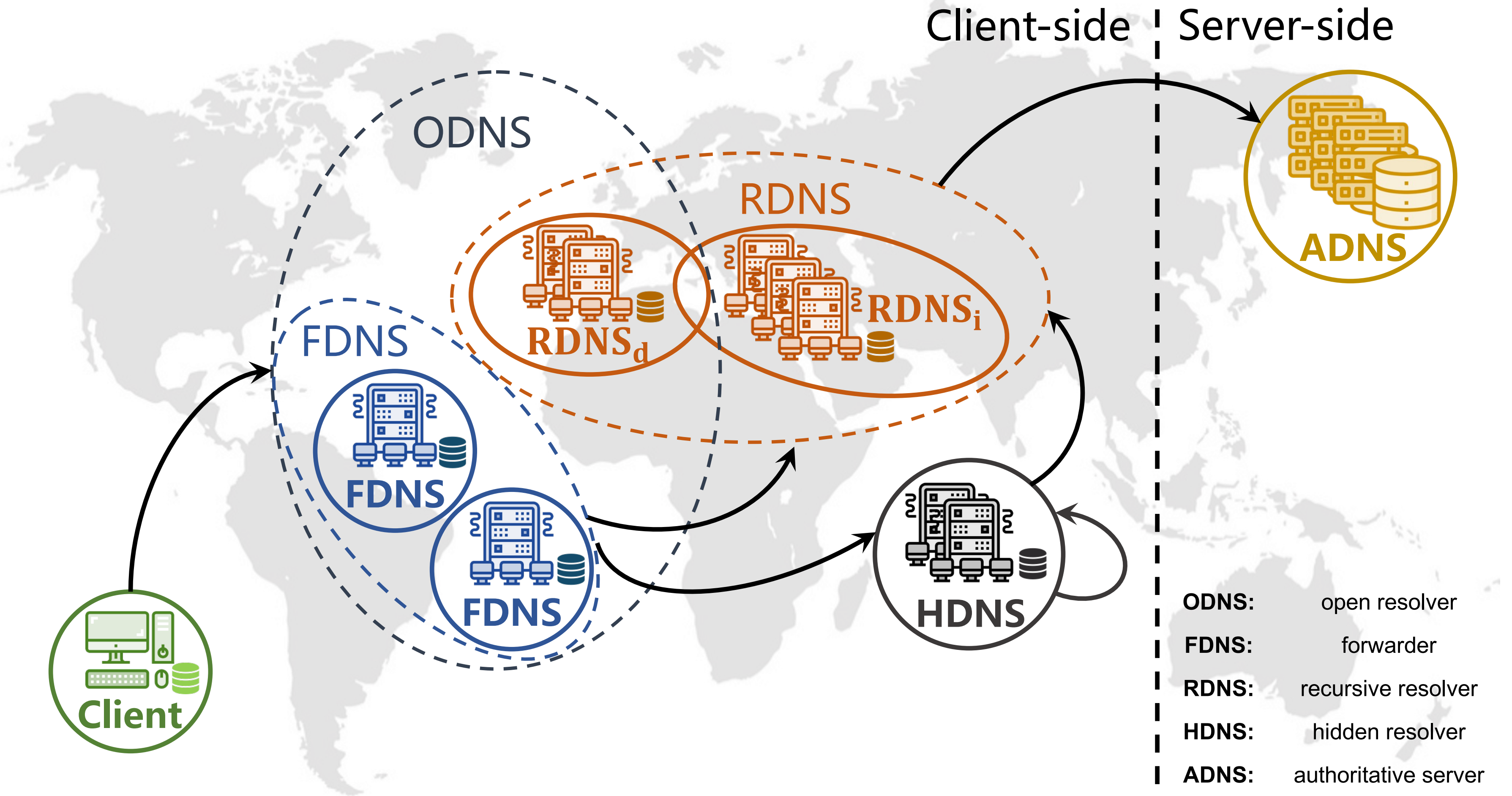
Question

What is the CDNS resolver?

A **conditional DNS resolver** with both **recursive** and **forwarding** query modes.

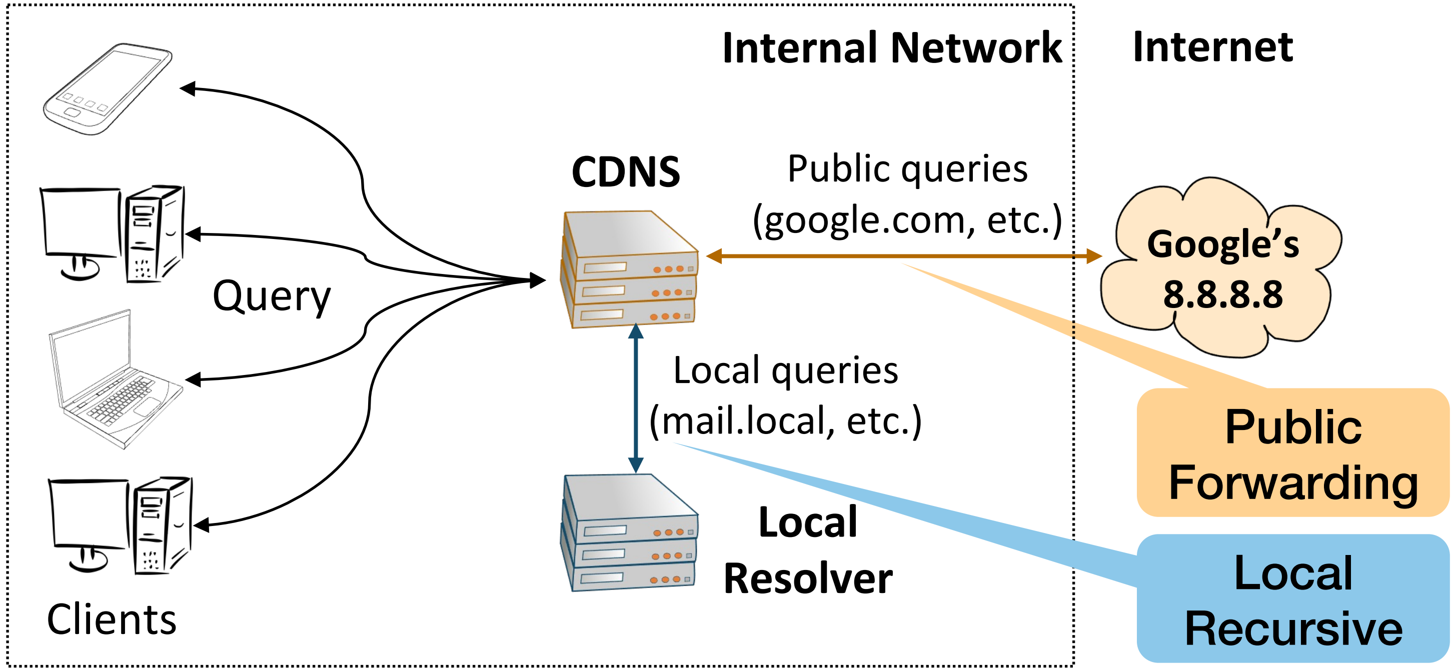
DNS Resolvers

- **Worldwide**
- **Multiple Roles**
  - ❑ Recursive, forwarder
  - ❑ Hidden DNS (HDNS)
- **Complex Interacting**
- **CDNS**
  - ❑ One of HDNSes
  - ❑ Never been studied



Attack Target: CDNS

- **Conditional DNS Resolver (CDNS)**
  - Forwarder + recursive resolver (shared cache)
  - 2 query zones used for different resolution
    - Z_F: domains for forwarding queries
    - Z_R: domains for recursive queries
- **Usage Scenarios**
  - Enterprise: splitting networks
  - ISP: reducing heavy traffic cost
  - (video-style domains)



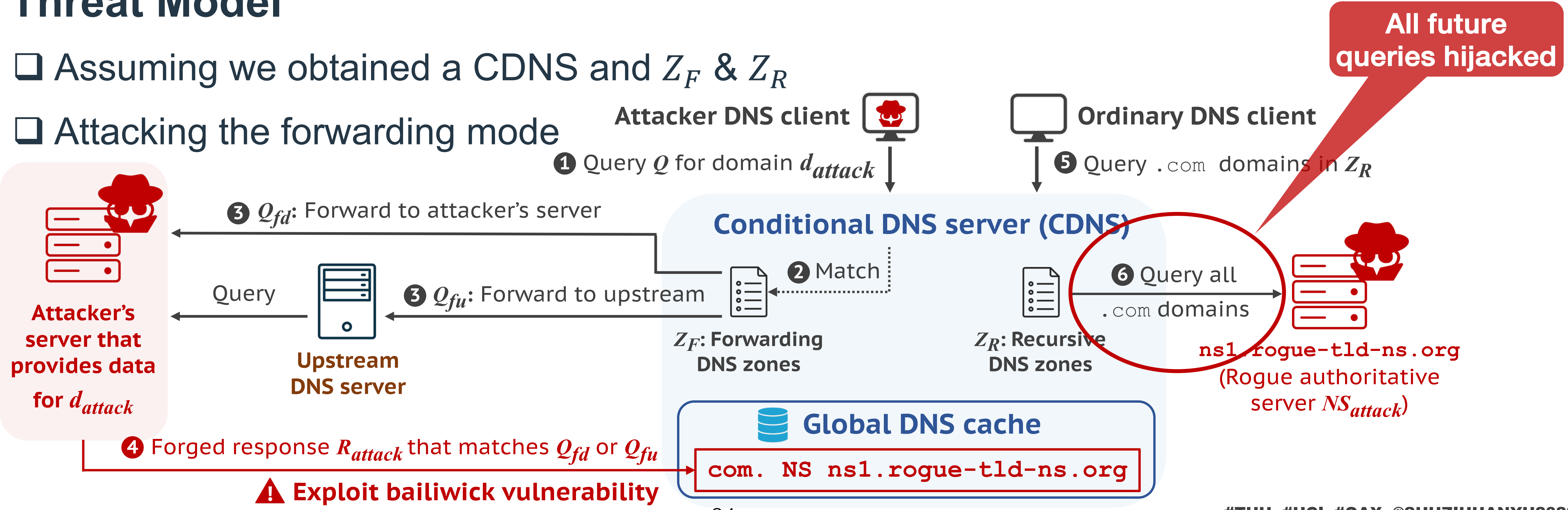
Attack Overview of MaginotDNS

➢ Attack Target

- ❑ CDNS that can be accessed

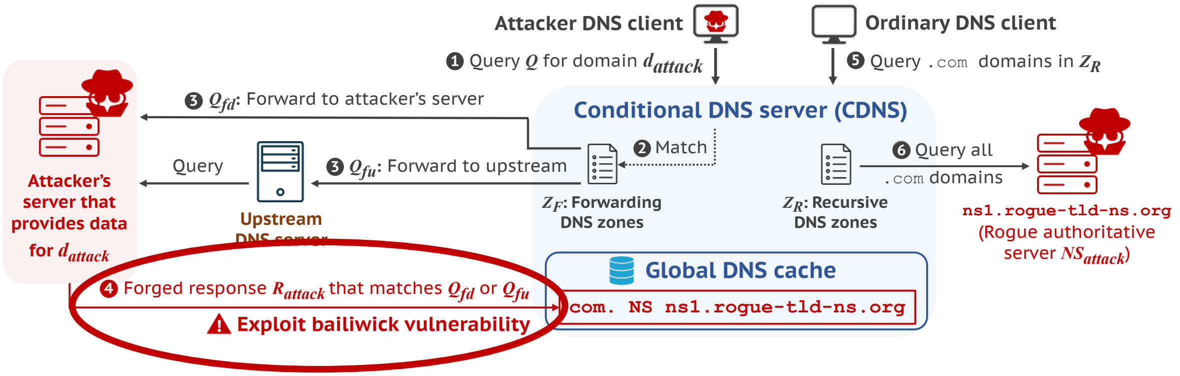
➢ Threat Model

- ❑ Assuming we obtained a CDNS and Z_F & Z_R
- ❑ Attacking the forwarding mode



Attack Overview of MaginotDNS

- **Bailiwick Checking Vulnerability**
  - ❑ In the forwarding mode
  - ❑ **Accepting all records in a forwarding res.**

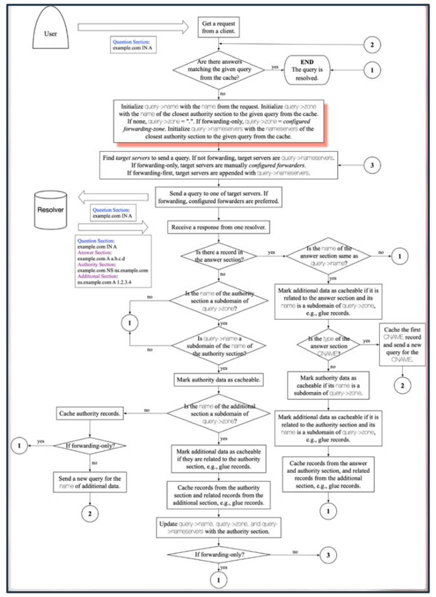


- **Exploiting Idea**
  - ❑ Bailiwick checking of the recursive mode is **well implemented**
  - ❑ But the **forwarding** mode is not.
  - ❑ Since they share the **same global DNS cache**
  - ❑ We can **exploit the weak forwarder** to attack the well-protected recursive
    - → **Breaking the boundary of DNS caching protection**

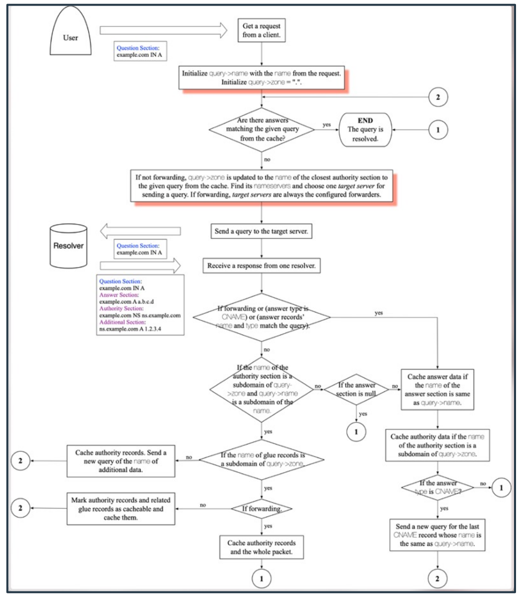
Software Analysis

➢ Finding Vulnerable Software

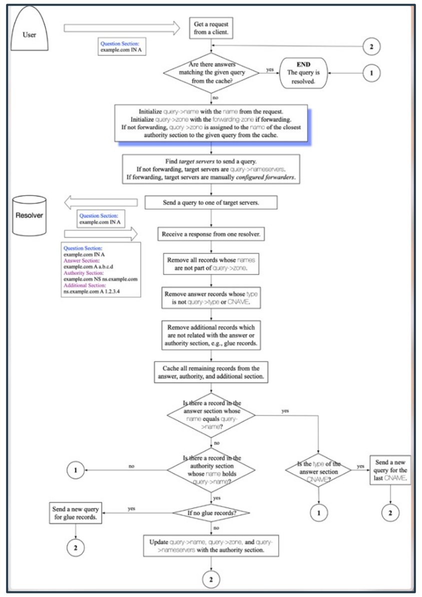
- ❑ In depth **bailiwick checking implementation** analysis
- ❑ Via source code review, debugging, and testing
- ❑ 8 mainstream DNS software, e.g., BIND and Microsoft DNS



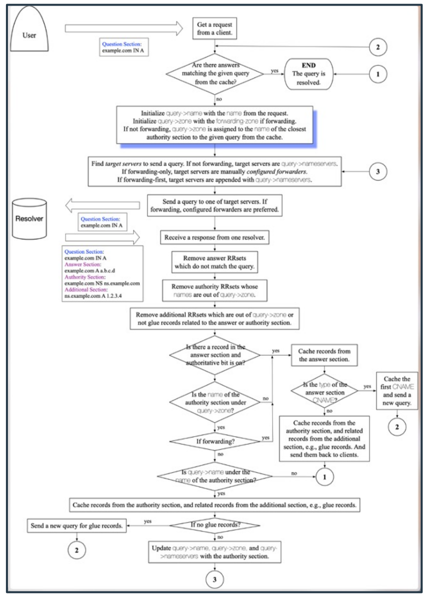
BIND



Knot



PowerDNS



Unbound

Extracting bailiwick checking implementations

Root Cause & Vulnerable Software

➤ General Bailiwick Checking Logic

❑ Summarized by us

➤ Root Cause

❑ In the `InitQuery` function:

- `Qry.zone` is set to root → all records is **in-bailiwick** (root's subdomains)

➤ Vulnerable Software

DNS Software	Forwarding	Recursive	Vulnerable
BIND9	Enabled	Enabled	**Yes**
Knot Resolver	Enabled	Enabled	**Yes**
Microsoft DNS	Enabled	Enabled	**Yes**
Technitium	Enabled	Enabled	**Yes**

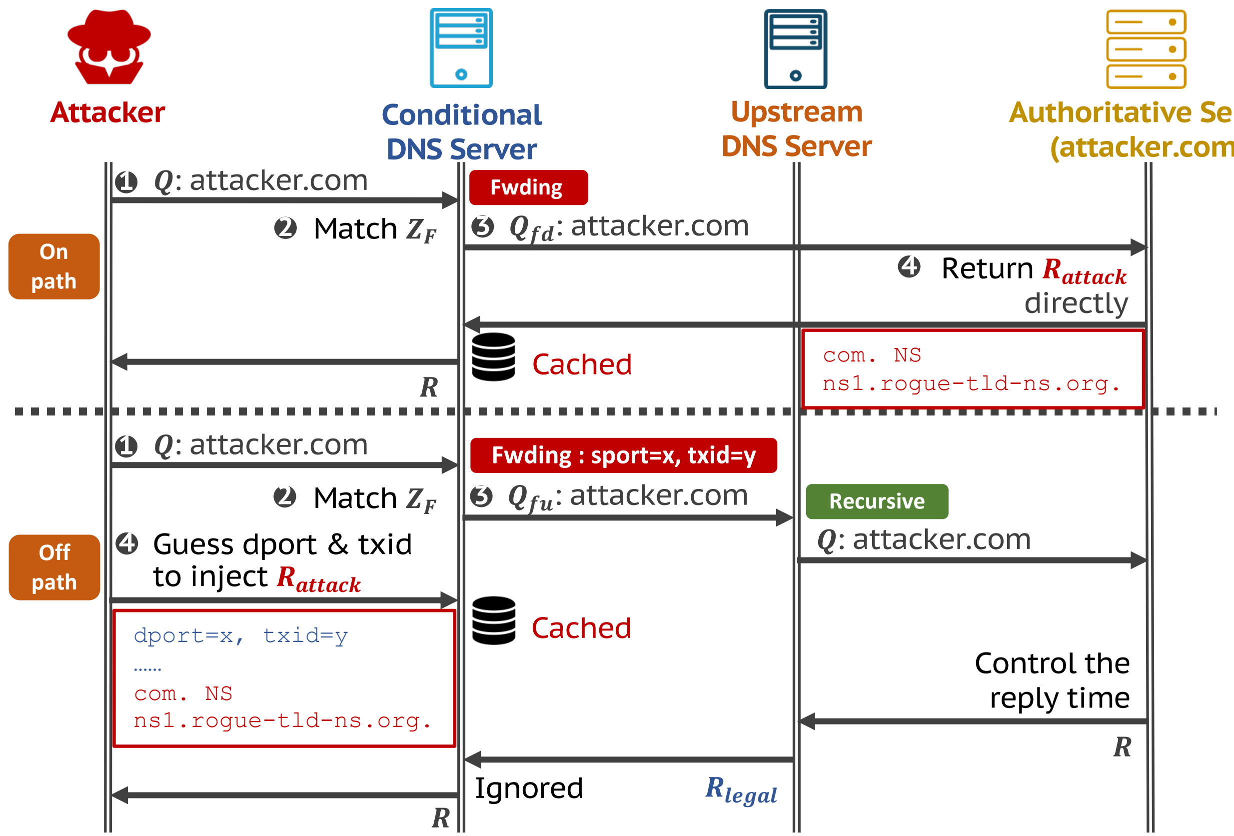
Algorithm 1: DNS resolution process

```
input  : A DNS Request from clients
output : A DNS Reply to clients
main()
    step_0: InitQuery(Q, Request)
    step_1: if SeachCache(Q, Cache) then
        goto final
    step_2: FindServers(Q, TgtSvrs)
    step_3: SendQuery(Q, TgtSvrs)
    step_4: ProcessResponse(Q, R)
        if ServerIsError(Q, R) then
            goto step 3
        if not MatchQuery(Q, R) then
            goto final
        SanitizeRecords(Q, R)
        if IsReferral(Q, R) then
            if not IsFwding() then
                UpdateQuery(Q)
                goto step 2
        if IsCNAME(Q, R) then
            UpdateQuery(Q)
            goto step 1
        CacheRecords(R, Cache)
    final: ConstructReply(Reply)
    return Reply
InitQuery(Q, Request)
    initialize Q.name, Q.type, Q.zone
    if IsFwding() then
        ModifyFwdQuery(Q)
SanitizeRecords(Q, R)
    for RR ∈ R do
        if OutofBailiwick(RR) then
            remove RR from R
UpdateQuery(Q, R)
    update Q.name, Q.type, Q.zone
```

Attack Steps of MaginotDNS

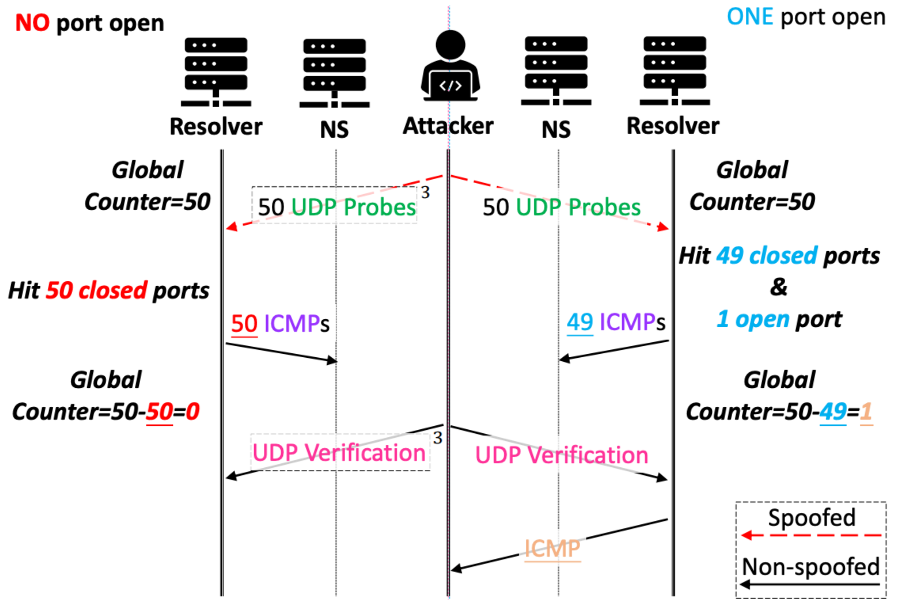
- **On-path Attack**
  - ❑ Returning fake responses directly
  - ❑ **BIND**, **MS DNS**, **Knot**, and **Technitium**
- **Off-path Attack**
  - ❑ Guessing src port & TXID with birthday attack
  - ❑ **Microsoft**: our found **new port vulnerability**
  - ❑ **BIND9**: extending the SADDNS attack

All future queries will be hacked.



Off-path Attack on BIND9

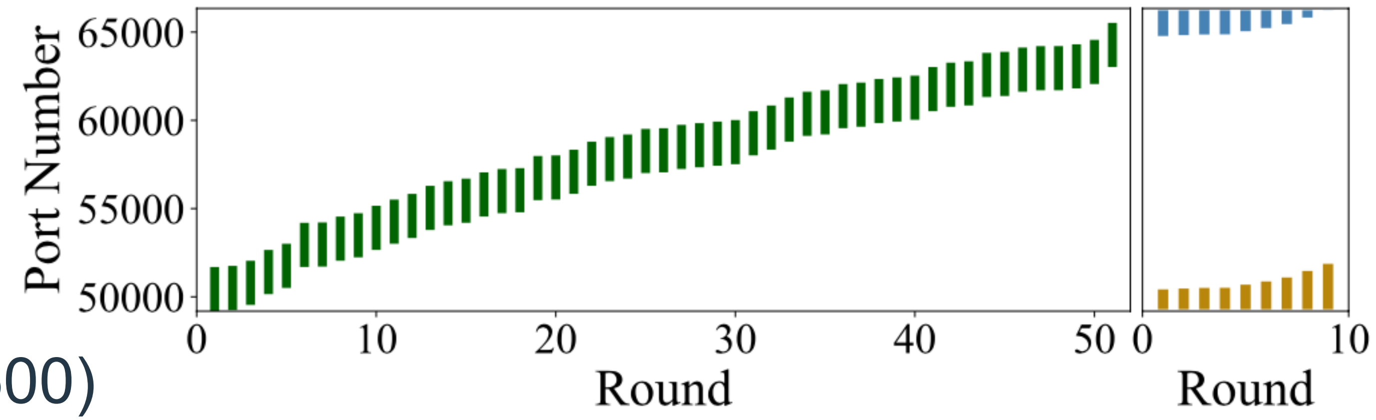
- **Guessing Source Port**
  - ❑ We use SADDNS to infer the source port
  - ❑ Only the in-use port is in the open state, while the others in the close state
  - ❑ ICMP rate-limit side-channel (check the SADDNS paper for details)
- **Brute-forcing TXID**
- **What We did**
  - ❑ Source port range: 32,768 - 60,999 (28,232)
  - ❑ Query timeout: 1.2s, guessing 50 ports each round
  - ❑ **Success rate** after 3,600 rounds:
    - $1 - [(28{,}232 - 50)/28{,}232]^{3{,}600} = 99.8\%$



SADDNS https://www.saddns.net/

Off-path Attack on Microsoft DNS

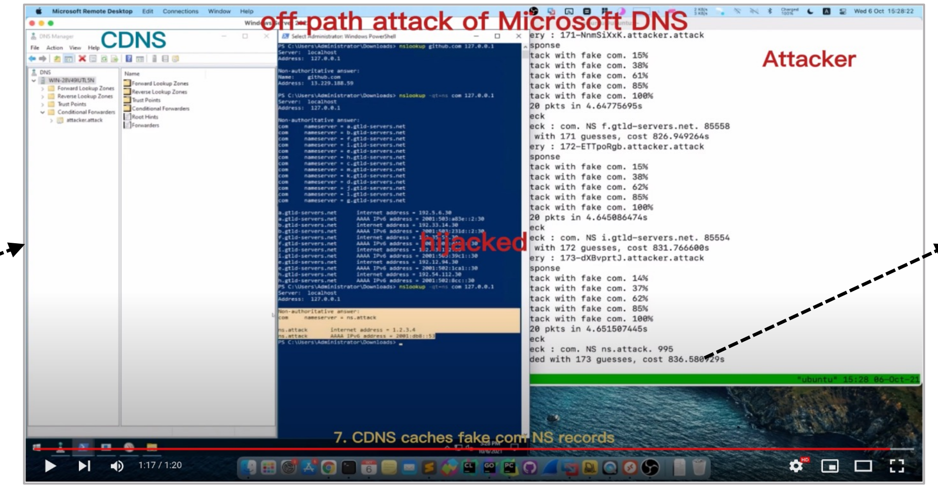
- **Guessing Source Port**
  - We found MS DNS only uses **~2,500 source ports** for resolution
  - 2,500 ports are **all in the open state** (SADDNS not working)
  - **Brute-forcing** all 2,500 ports
- **Brute-forcing TXID**
- **What We did**
  - Source port range: probing in advance (2,500)
  - Query timeout: 5s, guessing 20 ports each round
  - **Success rate** after 720 rounds:
    - $1 - [(2{,}500 - 20)/2{,}500]^{720} = 99.7\%$



Source Port Range Examples of Microsoft DNS

MaginotDNS Attack Demos

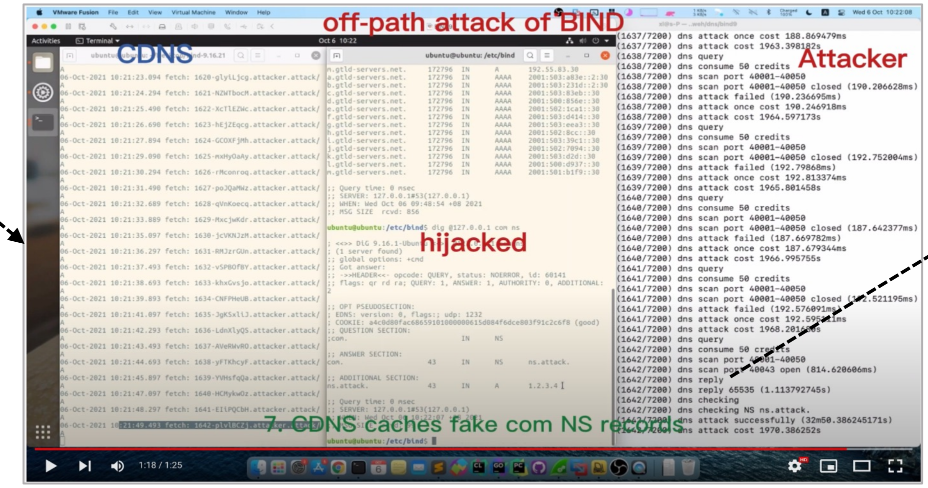
- ➤ **On-path Attack**
  - ❑ The result is determinative
- ➤ **Off-path Attack**
  - ❑ Microsoft: **avg. 802s**
  - ❑ BIND9: **avg. 790s**




```
Mon Aug  9 03:31:01 2021 : (2/360) dns query : 2-BatHkHSX.idealeer.com
Mon Aug  9 03:31:01 2021 : (2/360) dns response
Mon Aug  9 03:31:03 2021 : (2/360) dns attack with fake com. 15%
Mon Aug  9 03:31:04 2021 : (2/360) dns attack with fake com. 37%
Mon Aug  9 03:31:05 2021 : (2/360) dns attack with fake com. 60%
Mon Aug  9 03:31:06 2021 : (2/360) dns attack with fake com. 85%
Mon Aug  9 03:31:06 2021 : (2/360) dns attack with fake com. 100%
Mon Aug  9 03:31:06 2021 : to 202.112.238.57 : 1310720 pkts in 4.632276358s
Mon Aug  9 03:31:06 2021 : (2/360) dns check
Mon Aug  9 03:31:06 2021 : (2/360) dns check : com. NS gtld-servers.attack.
Mon Aug  9 03:31:06 2021 : dns attack succeeded with 2 guesses, cost 10.079395433s
```

Log of Attacking Microsoft

Watch videos here.

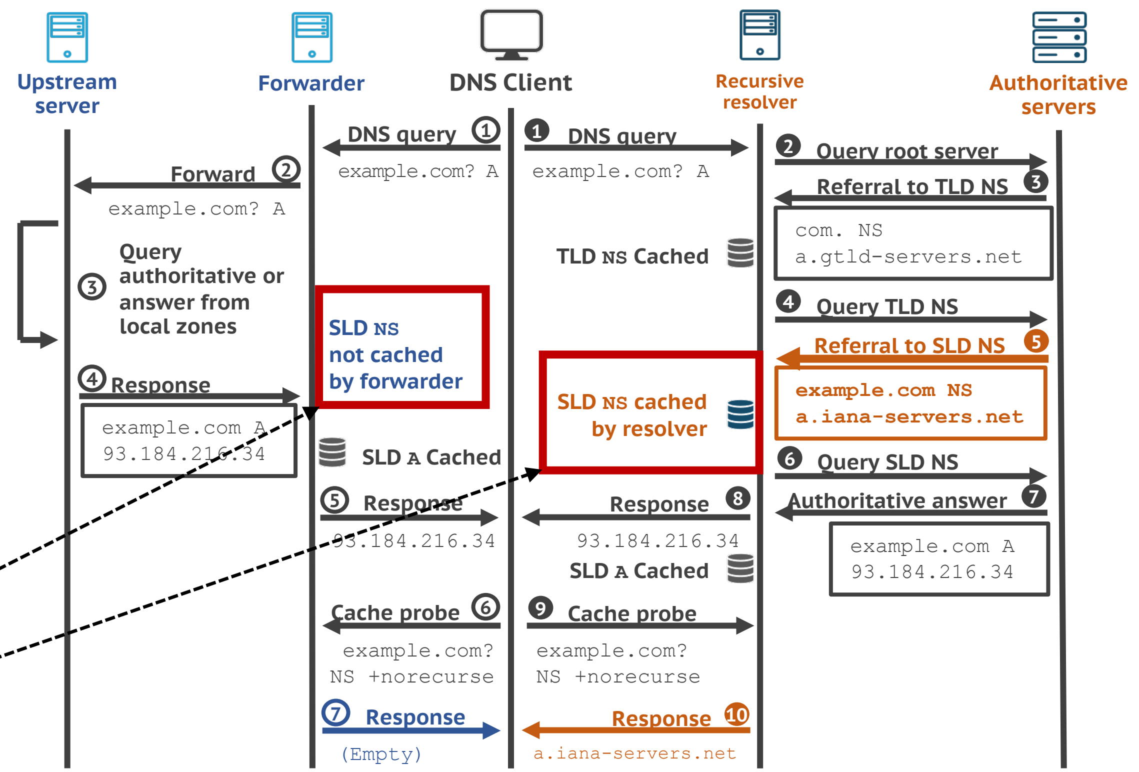



```
Thu Aug 26 23:10:53 2021 : (661/3600) dns querying
Thu Aug 26 23:10:53 2021 : (661/3600) dns consuming 50 credits
Thu Aug 26 23:10:53 2021 : (661/3600) dns scanning port 40001-40050
Thu Aug 26 23:10:54 2021 : (661/3600) dns scanning port 40020 open (651.902104ms)
Thu Aug 26 23:10:54 2021 : (661/3600) dns replying
Thu Aug 26 23:10:54 2021 : (661/3600) dns replying 65535 (928.938966ms)
Thu Aug 26 23:10:54 2021 : (661/3600) dns checking
Thu Aug 26 23:10:54 2021 : (661/3600) dns checking NS gtld-servers.attack.
Thu Aug 26 23:10:54 2021 : (661/3600) dns attack successfully (13m12.992182401s)
Thu Aug 26 23:10:54 2021 : (661/3600) dns attack cost (13m12.99219492s)
```

Log of Attacking BIND9

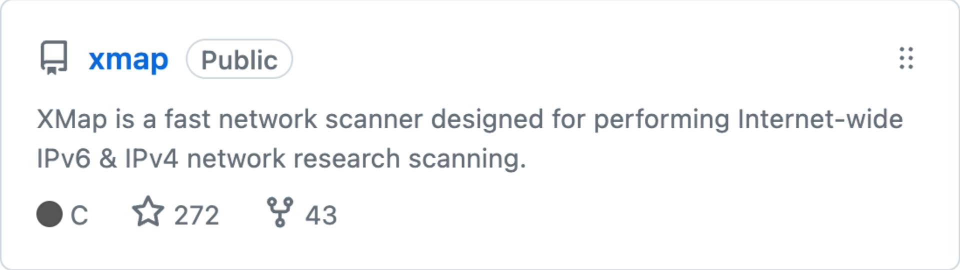
Finding Vulnerable CDNSes

- **Differentiating Forwarder & Recursive**
  - Based on the DNS resolution mechanism
  - **Forwarders** do not cache **intermediate NS records**
- **Finding CDNSes**
  - New methodology
    1. Targeting one resolver
    2. Testing a group of domains, sending **NS&NR** queries
    3. For some domains, no NS responses (**forwarding**)
    4. For others, we get NS responses (**recursive**)
    5. The resolver does **both forwarding & recursive resolution**
    6. → **CDNS identified**



Vulnerable CDNS Population

➢ Measurement with XMap



- ❑ We collected **1.2M resolvers**
- ❑ Removing not-applicable ones, such as violating NR or multiple caches
- ❑ Applying our **new method** to identify **154,955 CDNSes**
- ❑ Using **software fingerprints** to locate **54,949 vulnerable CDNSes**
  - Resolvers with DNSSEC or 0x20 are filtered out

CDNSes identified by probing	**154,955**	**41.8%**
– Version identifiable (in CDNS)	117,306	31.7%
– by `version.bind`	59,419	16.0%
– by `fpdns`	57,887	15.6%
– OS identified for BIND (in CDNS)	19,995	5.4%
– DNSSEC validation (in CDNS)	34,424	9.3%
– 0x20 encoding (in CDNS)	1,119	0.3%

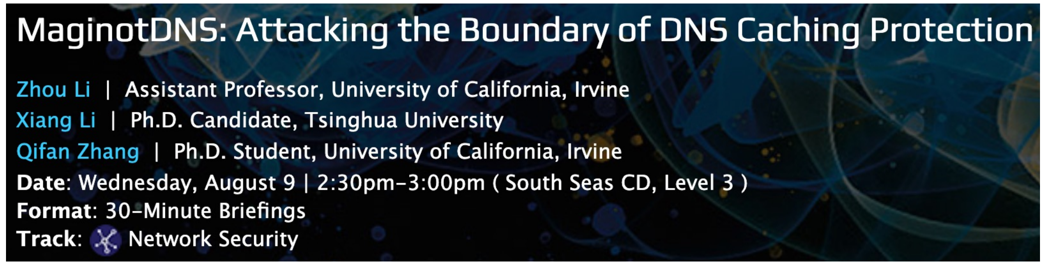
Vulnerable CDNSes	**54,949**	**14.8%**
– On-path attack possible*	**54,949**	**14.8%**
– BIND	24,287	6.6%
– Microsoft DNS	30,662	8.3%
– Off-path attack possible*	**48,539**	**13.1%**
– BIND (OS exploitable)	17,877	4.8%
– Microsoft DNS	30,662	8.3%
– Recursive-default	10,445	5.0%
– Forwarding-default	36,581	9.9%

Discussion & Mitigation

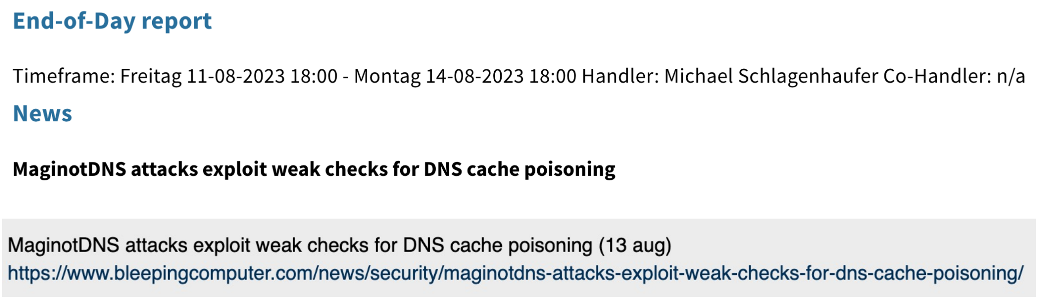
- **Vulnerability Disclosure**
  - Confirmed and fixed by **all affected software**: BIND9, Knot, Microsoft, & Technitium
  - **4 CVE-ids** published & **Bounty** awarded by Microsoft
- **Root Cause**
  - Poor forwarding bailiwick checking implementation
    - `Qry.zone` is set to root → all records is **in-bailiwick** (root's subdomains)
- **Mitigation Solution**
  - `Qry.zone` should be set to the forwarded domain in Z_F
  - Then only records under forwarded domain are acceptable
  - Have been adopted by affected software

Real-world Impact

- ➢ **Industry**
  - ❑ Presented at Black Hat USA 2023
- ➢ **Government/University**
  - ❑ An Austria government CERT daily report
  - ❑ A Sweden government CERT weekly news
  - ❑ A Bournemouth University (BU) CERT news
- ➢ **60+ News Coverage**
  - ❑ E.g., BleepingComputer
- ➢ **APNIC Blog**


MaginotDNS: Attacking the Boundary of DNS Caching Protection

Zhou Li | Assistant Professor, University of California, Irvine
Xiang Li | Ph.D. Candidate, Tsinghua University
Qifan Zhang | Ph.D. Student, University of California, Irvine
Date: Wednesday, August 9 | 2:30pm–3:00pm (South Seas CD, Level 3)
Format: 30-Minute Briefings
Track: Network Security

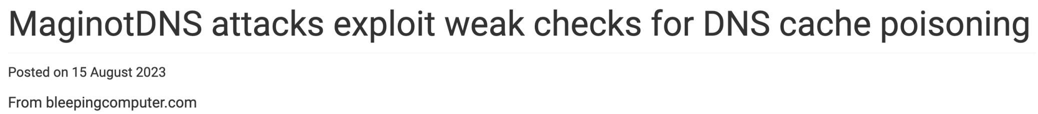

End-of-Day report

Timeframe: Freitag 11-08-2023 18:00 - Montag 14-08-2023 18:00 Handler: Michael Schlagenhaufer Co-Handler: n/a

News

MaginotDNS attacks exploit weak checks for DNS cache poisoning

MaginotDNS attacks exploit weak checks for DNS cache poisoning (13 aug)
https://www.bleepingcomputer.com/news/security/maginotdns-attacks-exploit-weak-checks-for-dns-cache-poisoning/


MaginotDNS attacks exploit weak checks for DNS cache poisoning

Posted on 15 August 2023

From bleepingcomputer.com


MaginotDNS attacks exploit weak checks for DNS cache poisoning

By Bill Toulas August 13, 2023 10:12 AM 0

Conclusion

- **New Threat Model**
  - ❑ A new resolver role: CDNS
- **New Attack Surface, Vulnerabilities, & Attacks**
  - ❑ Mixed roles and shared cache
  - ❑ Inconsistency of DNS implementation
  - ❑ Old DNS mechanism
  - ❑ New Vulnerabilities & Attacks
- **New Methodology & Results**
  - ❑ CDNS identifying method
  - ❑ Numbers of vulnerable CDNSes

Wrap-up

Thanks for listening!
Any questions?

Xiang Li, Tsinghua University

x-l19@mails.tsinghua.edu.cn

Paper

Tool