Automated Discovery of DNS Resolver Vulnerabilities with Stateful Fuzzing

Qifan Zhang, Xuesong Bai, Xiang Li, Haixin Duan, Qi Li and Zhou Li

DINR 2024 talk

04/04/2024

UCI Samueli School of Engineering

University of California, Irvine

DNS Failures & Attacks Happened a Lot

Help Net Security

October 26, 2021

72% of organizations hit by DNS attacks in the past year

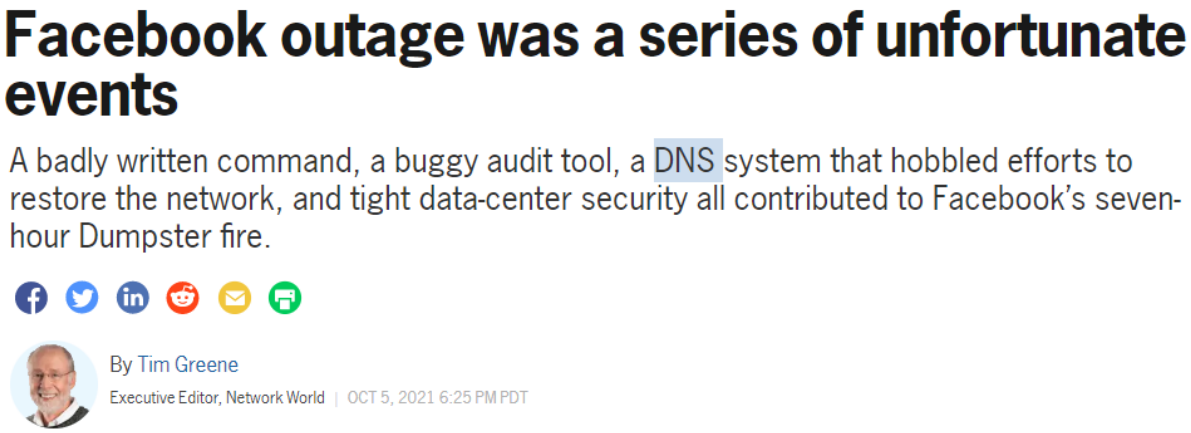
Unpatched DNS Bug Puts Millions of Routers, IoT Devices at Risk

MASQUERADE PARTY —

DNS cache poisoning, the Internet attack from 2008, is back from the dead

A newly found side channel in a widely used protocol lets attackers spoof domains.

DAN GOODIN - 11/12/2020, 6:30 AM

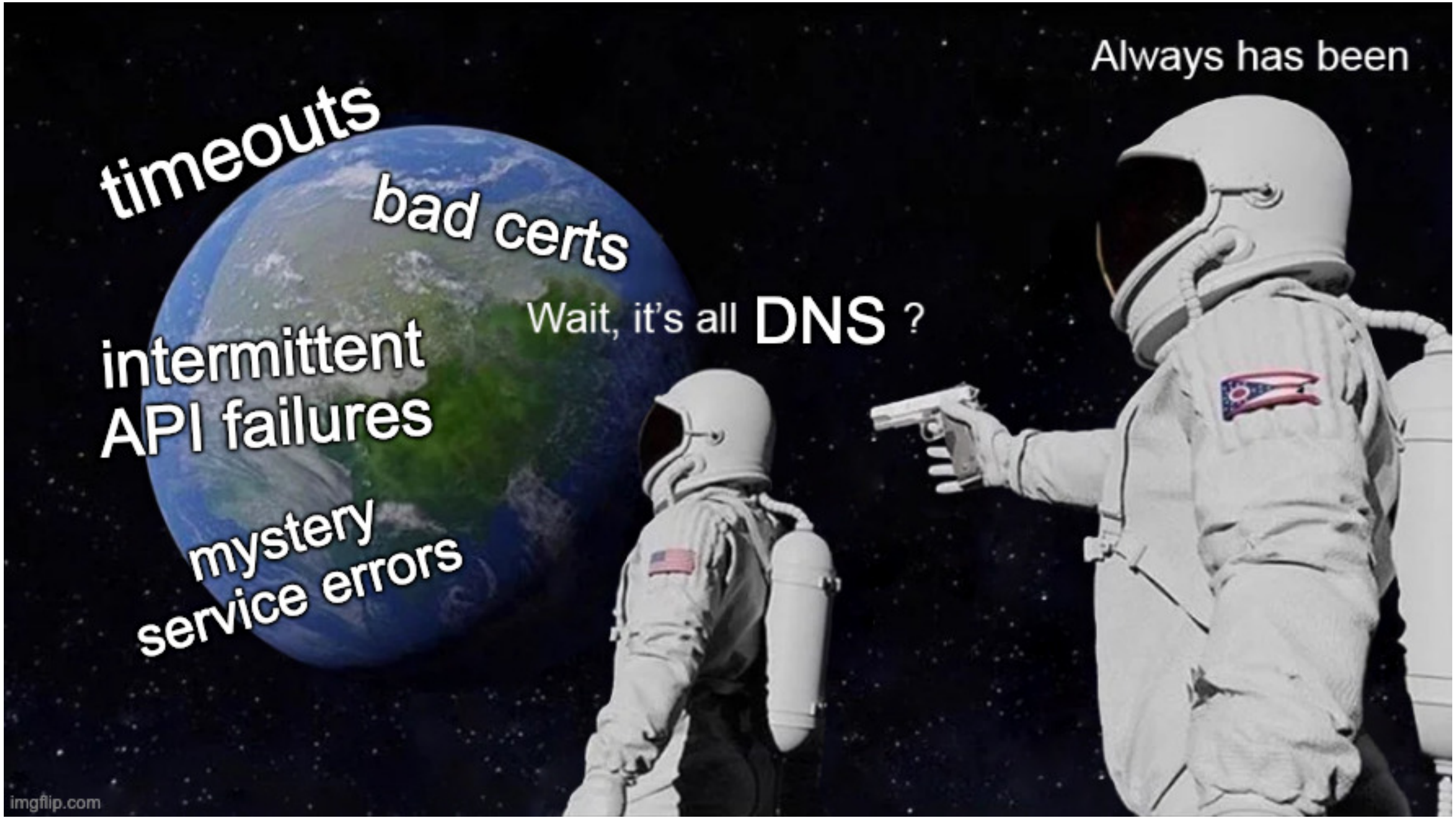


Facebook outage was a series of unfortunate events

A badly written command, a buggy audit tool, a DNS system that hobbled efforts to restore the network, and tight data-center security all contributed to Facebook's seven-hour Dumpster fire.

By Tim Greene

Executive Editor, Network World | OCT 5, 2021 6:25 PM PDT


timeouts
bad certs
intermittent API failures
mystery service errors
Wait, it's all DNS ?
Always has been
imgflip.com

Outline

- Automated discovery of DNS bugs with fuzzing
  - ResolverFuzz [Security'24]
- Configuration-guided fuzzing
  - Ongoing work
- Conclusion

https://dns-debug.github.io/

Fuzzing in a Nutshell

```
$ ./testme --help
Usage: testme <int32_arg>

$ ./testme AAAA
Please enter an integer!

$ cat fuzzer.sh
while :
do
  input="$(dd if=/dev/urandom bs=4 count=1)"
  ./testme $input || echo $input >> crash_seeds
done
```

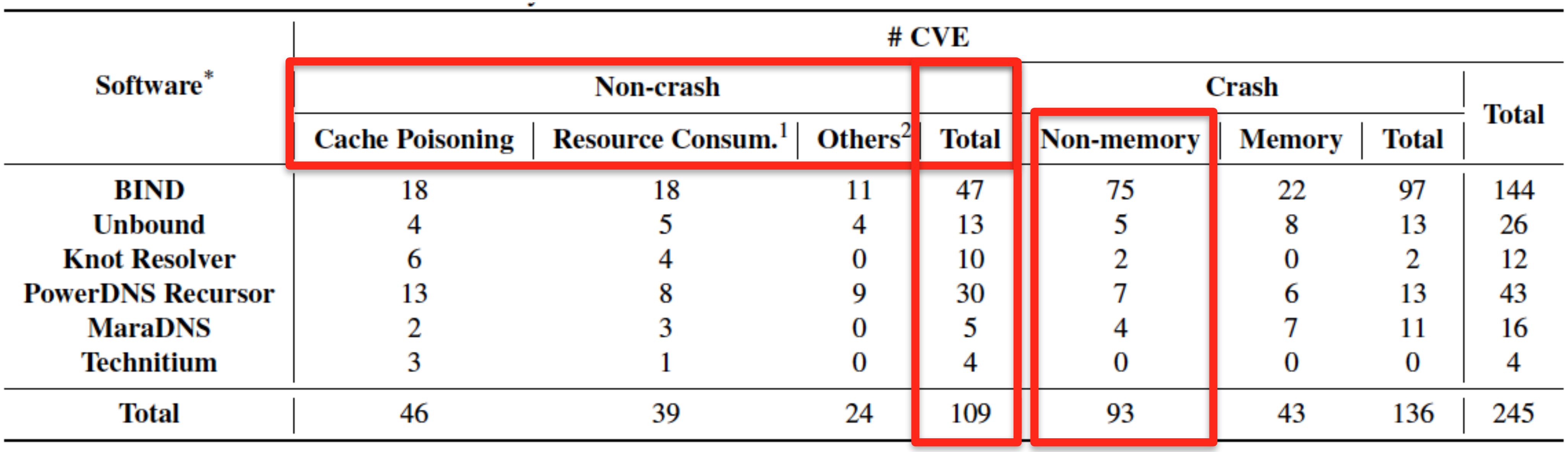
Slides credit: Mathias Payer

Challenges of DNS Fuzzing

- Standard Fuzzing
  - Stateless (program reset after 1 input)
  - Default configuration
  - Focusing on software crash
  - Single programming language
- DNS fuzzing
  - Stateful (query & response, resolver cache)
  - Customized configurations
  - Crash, cache poisoning, denial of service, …
  - Multilingual system (C, C++, C#, Go)

DNS CVEs

- Manual analysis of *423* DNS CVEs from 1999-2023
  - *291* CVEs about 6 DNS software
    - *245* CVEs about DNS resolvers
      - *109* CVEs don’t trigger any crash!
      - *93* crash CVEs are non-memory (e.g., assertion failures)



Software*	# CVE							
	Non-crash				Crash			Total
	Cache Poisoning	Resource Consum.[1]	Others[2]	Total	Non-memory	Memory	Total	
BIND	18	18	11	47	75	22	97	144
Unbound	4	5	4	13	5	8	13	26
Knot Resolver	6	4	0	10	2	0	2	12
PowerDNS Recursor	13	8	9	30	7	6	13	43
MaraDNS	2	3	0	5	4	7	11	16
Technitium	3	1	0	4	0	0	0	4
Total	46	39	24	109	93	43	136	245

ResolverFuzz [Security’24]

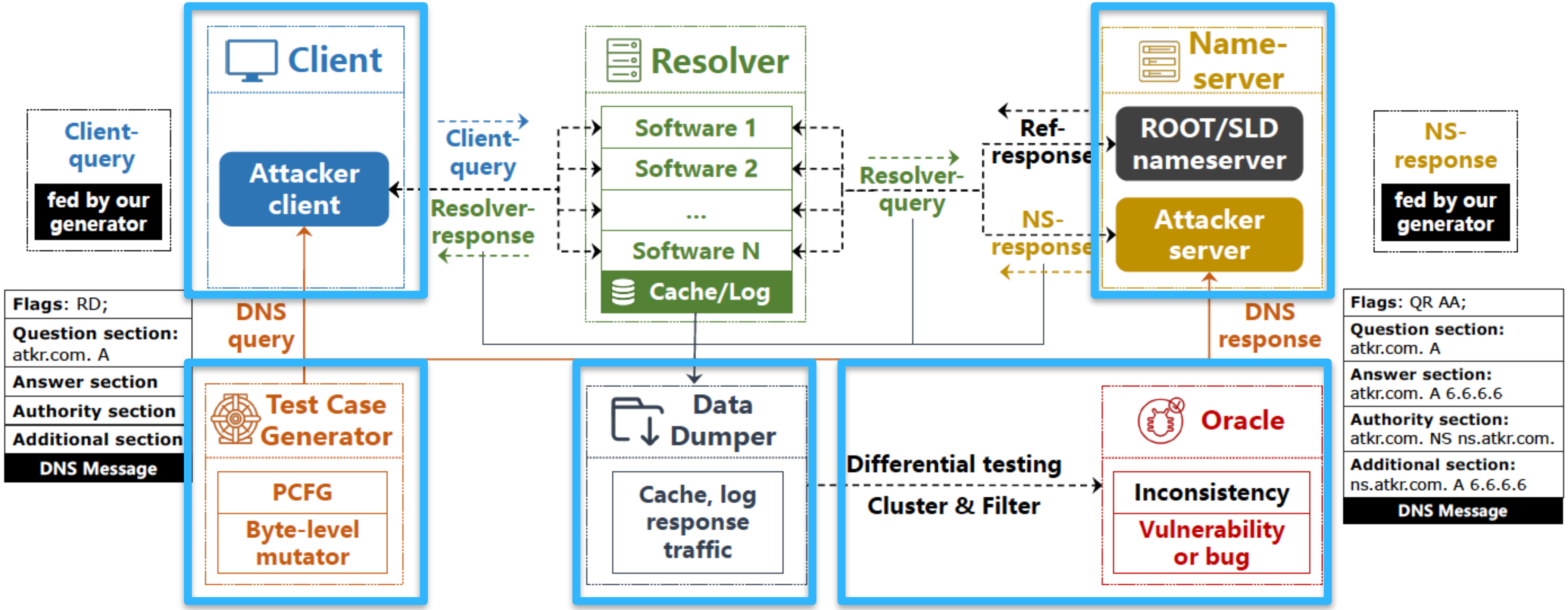


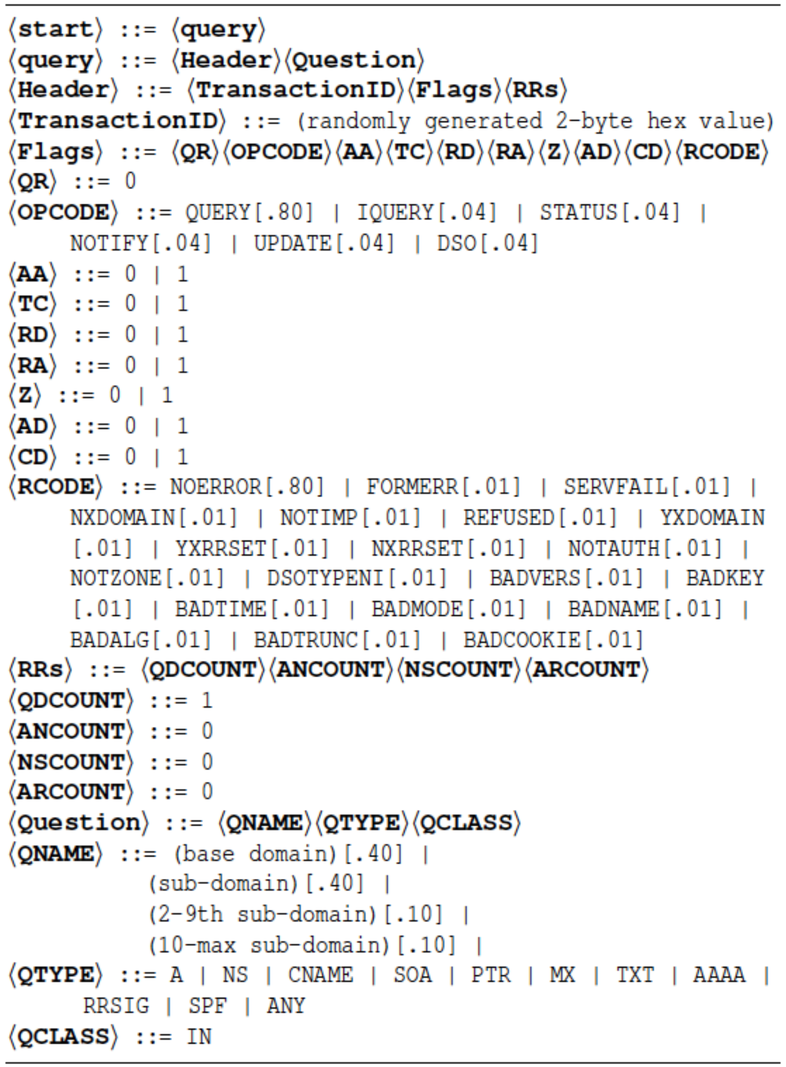
Figure 3: Workflow of RESOLVERFUZZ.

[Security’24] Zhang et al. ResolverFuzz: Automated Discovery of DNS Resolver Vulnerabilities with Query-Response Fuzzing

ResolverFuzz: Test Case Generation

- PCFG (Probabilistic Context-Free Grammar)
  – Probability assignment based on CVE study
  – Following DNS syntax, no semantics
- Byte mutation
  – Special characters to trigger decoding issue
    – \., \000, @, /, \, … [1]

[1] Jeitner et al., Injection Attacks Reloaded: Tunnelling Malicious Payloads over DNS. Usenix Security’21.

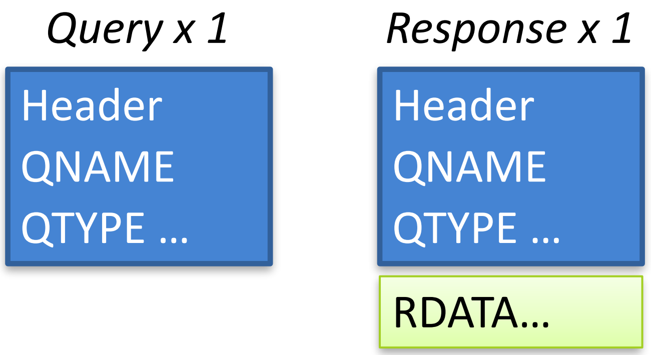


```
⟨start⟩ ::= ⟨query⟩
⟨query⟩ ::= ⟨Header⟩⟨Question⟩
⟨Header⟩ ::= ⟨TransactionID⟩⟨Flags⟩⟨RRs⟩
⟨TransactionID⟩ ::= (randomly generated 2-byte hex value)
⟨Flags⟩ ::= ⟨QR⟩⟨OPCODE⟩⟨AA⟩⟨TC⟩⟨RD⟩⟨RA⟩⟨Z⟩⟨AD⟩⟨CD⟩⟨RCODE⟩
⟨QR⟩ ::= 0
⟨OPCODE⟩ ::= QUERY[.80] | IQUERY[.04] | STATUS[.04] |
    NOTIFY[.04] | UPDATE[.04] | DSO[.04]
⟨AA⟩ ::= 0 | 1
⟨TC⟩ ::= 0 | 1
⟨RD⟩ ::= 0 | 1
⟨RA⟩ ::= 0 | 1
⟨Z⟩ ::= 0 | 1
⟨AD⟩ ::= 0 | 1
⟨CD⟩ ::= 0 | 1
⟨RCODE⟩ ::= NOERROR[.80] | FORMERR[.01] | SERVFAIL[.01] |
    NXDOMAIN[.01] | NOTIMP[.01] | REFUSED[.01] | YXDOMAIN
    [.01] | YXRRSET[.01] | NXRRSET[.01] | NOTAUTH[.01] |
    NOTZONE[.01] | DSOTYPENI[.01] | BADVERS[.01] | BADKEY
    [.01] | BADTIME[.01] | BADMODE[.01] | BADNAME[.01] |
    BADALG[.01] | BADTRUNC[.01] | BADCOOKIE[.01]
⟨RRs⟩ ::= ⟨QDCOUNT⟩⟨ANCOUNT⟩⟨NSCOUNT⟩⟨ARCOUNT⟩
⟨QDCOUNT⟩ ::= 1
⟨ANCOUNT⟩ ::= 0
⟨NSCOUNT⟩ ::= 0
⟨ARCOUNT⟩ ::= 0
⟨Question⟩ ::= ⟨QNAME⟩⟨QTYPE⟩⟨QCLASS⟩
⟨QNAME⟩ ::= (base domain)[.40] |
          (sub-domain)[.40] |
          (2-9th sub-domain)[.10] |
          (10-max sub-domain)[.10] |
⟨QTYPE⟩ ::= A | NS | CNAME | SOA | PTR | MX | TXT | AAAA |
     RRSIG | SPF | ANY
⟨QCLASS⟩ ::= IN
```

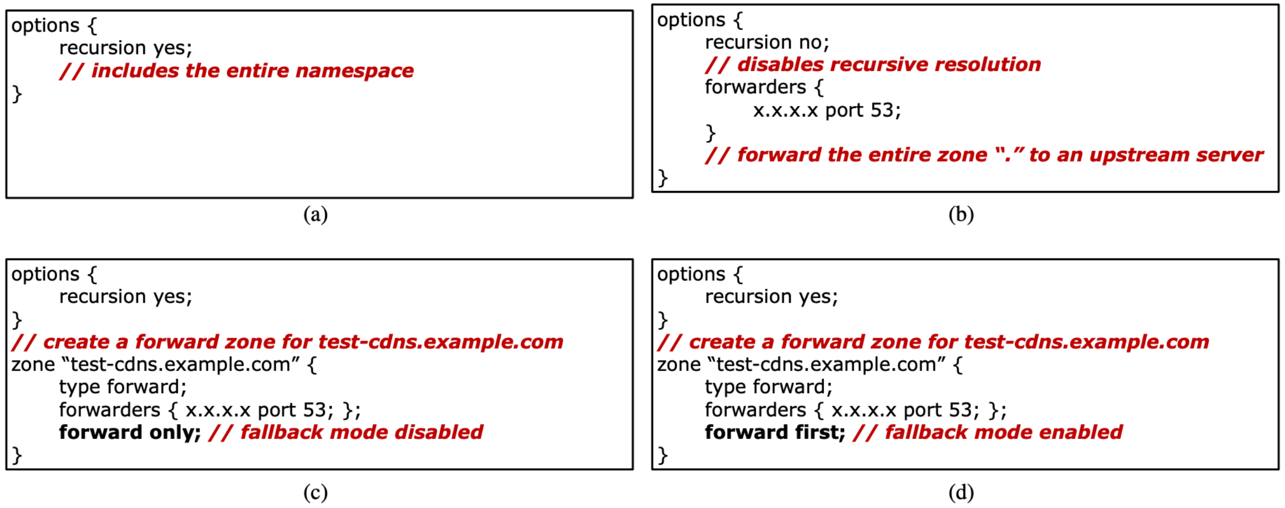
Listing 1: PCFG for DNS query.

ResolverFuzz: Stateful Fuzzing

- Query-response input
  - Short sequence based on CVE study



- Selective configurations
  - Recursive-only
  - Forward-only
  - Conditional DNS (CDNS)
  - CDNS with fallback



```
options {
      recursion yes;
      // includes the entire namespace
}
```

(a)

```
options {
      recursion no;
      // disables recursive resolution
      forwarders {
            x.x.x.x port 53;
      }
      // forward the entire zone "." to an upstream server
}
```

(b)

```
options {
      recursion yes;
}
// create a forward zone for test-cdns.example.com
zone "test-cdns.example.com" {
      type forward;
      forwarders { x.x.x.x port 53; };
      forward only; // fallback mode disabled
}
```

(c)

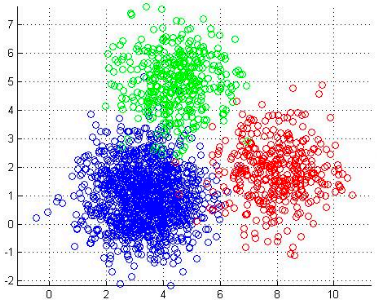
```
options {
      recursion yes;
}
// create a forward zone for test-cdns.example.com
zone "test-cdns.example.com" {
      type forward;
      forwarders { x.x.x.x port 53; };
      forward first; // fallback mode enabled
}
```

(d)

Figure 12: Example BIND configs of a) recursive-only, b) forward-only, c) CDNS without fallback, and d) CDNS with fallback.

ResolverFuzz: Oracles

- Cache poisoning oracle
  - – Differential testing
  - – Clustering on cache records



Bisecting K-means

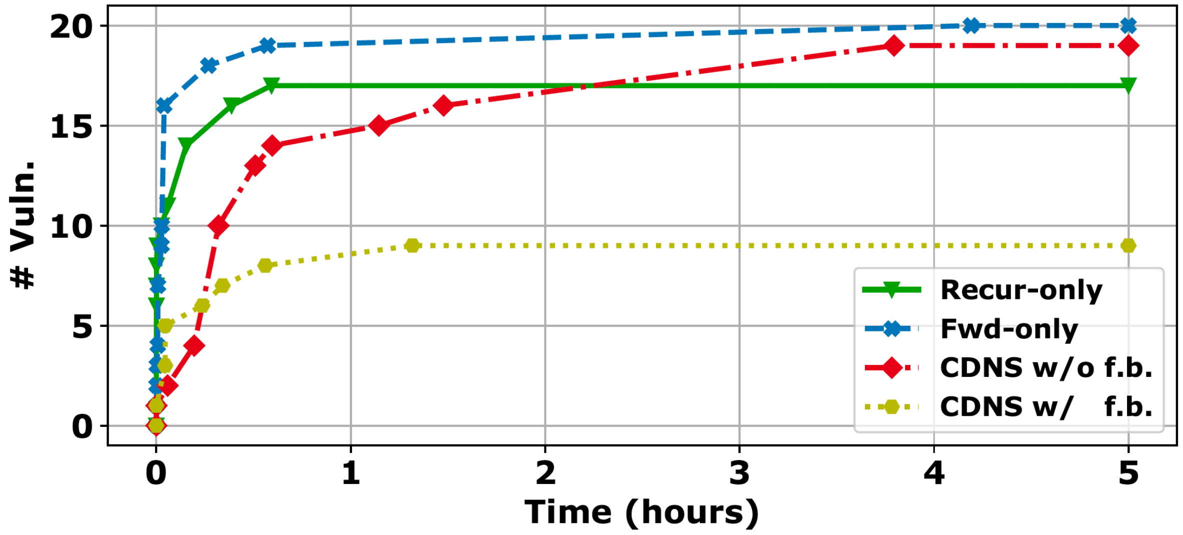
DNS Software cache records

- Resource consumption oracle
  - – Abnormal frequency of logged operations
    - – e.g., cache search

- Crash oracle
  - – Process monitoring in docker

Evaluation Results

- 23 bugs discovered
  - Cache poisoning, resource consumption, crash
  - 15 CVEs assigned
  - Outperform dns-fuzz-server, DNS fuzzer and SnapFuzz



(a) Recursive-only, forward-only and CDNS with/without fallback modes.

MaginotDNS [Security'23]   Phoenix Domain [NDSS'23]   TuDoor [S&P'24]

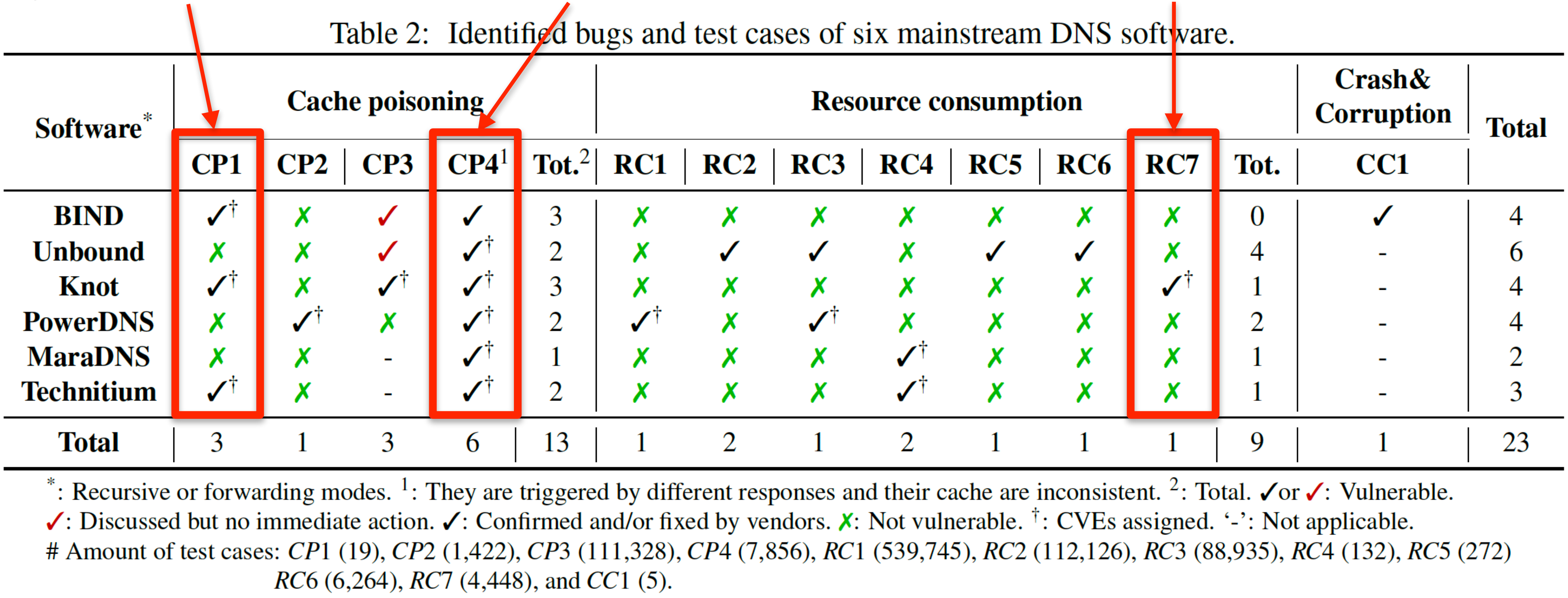


Table 2: Identified bugs and test cases of six mainstream DNS software.

Software*	Cache poisoning					Resource consumption								Crash& Corruption	Total
	CP1	CP2	CP3	CP4[1]	Tot.[2]	RC1	RC2	RC3	RC4	RC5	RC6	RC7	Tot.	CC1	
BIND	✓†	✗	✓	✓	3	✗	✗	✗	✗	✗	✗	✗	0	✓	4
Unbound	✗	✗	✓	✓†	2	✗	✓	✓	✗	✓	✓	✗	4	-	6
Knot	✓†	✗	✓†	✓†	3	✗	✗	✗	✗	✗	✗	✓†	1	-	4
PowerDNS	✗	✓†	✗	✓†	2	✓†	✗	✓†	✗	✗	✗	✗	2	-	4
MaraDNS	✗	✗	-	✓†	1	✗	✗	✗	✓†	✗	✗	✗	1	-	2
Technitium	✓†	✗	-	✓†	2	✗	✗	✗	✓†	✗	✗	✗	1	-	3
Total	3	1	3	6	13	1	2	1	2	1	1	1	9	1	23

*: Recursive or forwarding modes. [1]: They are triggered by different responses and their cache are inconsistent. [2]: Total. ✓or ✓: Vulnerable. ✓: Discussed but no immediate action. ✓: Confirmed and/or fixed by vendors. ✗: Not vulnerable. †: CVEs assigned. '-': Not applicable.

Amount of test cases: *CP*1 (19), *CP*2 (1,422), *CP*3 (111,328), *CP*4 (7,856), *RC*1 (539,745), *RC*2 (112,126), *RC*3 (88,935), *RC*4 (132), *RC*5 (272) *RC*6 (6,264), *RC*7 (4,448), and *CC*1 (5).

Outline

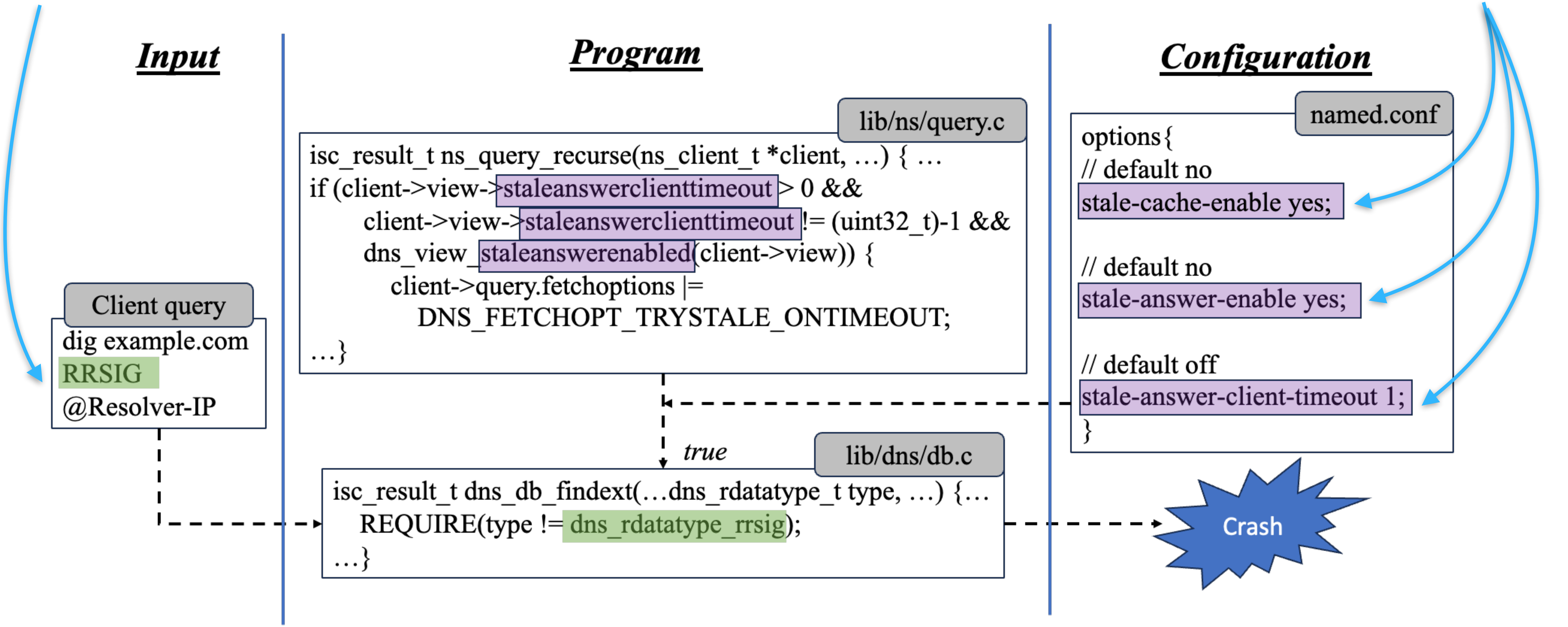
- Automated discovery of DNS bugs with fuzzing
  - ResolverFuzz [Security’24]
- Configuration-guided fuzzing
  - Ongoing work
- Conclusion

https://dns-debug.github.io/

A Configuration-related Bug

Normal DNSSEC Query

Customized config options



```
dig example.com
RRSIG
@Resolver-IP
```

```
isc_result_t ns_query_recurse(ns_client_t *client, …) { …
if (client->view->staleanswerclienttimeout > 0 &&
      client->view->staleanswerclienttimeout != (uint32_t)-1 &&
      dns_view_staleanswerenabled(client->view)) {
         client->query.fetchoptions |=
              DNS_FETCHOPT_TRYSTALE_ONTIMEOUT;
…}
```

```
isc_result_t dns_db_findext(…dns_rdatatype_t type, …) {…
   REQUIRE(type != dns_rdatatype_rrsig);
…}
```

```
options{
// default no
stale-cache-enable yes;

// default no
stale-answer-enable yes;

// default off
stale-answer-client-timeout 1;
}
```

BIND CVE-2022-3736 (CVSS: 7.5 High)

Challenges & Ideas & Plan

- Challenges for configuration-guided fuzzing
  - Large fuzzing space: *network input X configuration options*
  - Large rebooting overhead after changing configurations
  - Unknown syntax & semantics of *valid* configuration options
- Our ideas
  - Identifying *security-related* configuration options (e.g., CVE study)
  - *Collaborative* generation of seed input and configurations
  - Tracking *in-memory representations* of configuration options for rapid mutation
  - Fuzzing scheduling guided by configurations
- Evaluation plan
  - Profuzzbench (*10* out of *13* subjects support customized configurations)
  - DNS software (e.g., BIND9)

Outline

- Automated discovery of DNS bugs with fuzzing
  - ResolverFuzz [Security’24]
- Configuration-guided fuzzing
  - Ongoing work
- Conclusion

https://dns-debug.github.io/

Vulnerability Disclosure

- Vulnerable DNS software

BIND 9 KNOT RESOLVER unbound MaraDNS POWERDNS

Simple DNS Plus Microsoft DNS Technitium DNS Server

- Vulnerable public resolvers

Google Public DNS 1.1.1.1 ADGUARD DNS quad9 DYN

Final Thoughts

- DNS is a mature infrastructure, but still many problems
  - New RFCs, implementations, use cases
    - Old bugs can be revived!
  - Inconsistency & Under-specification
- Research questions
  - How to find more non-trivial DNS bugs?
    - Configuration-based stateful fuzzing (ongoing)
    - Longer sequence of requests and responses
  - “Universal” stateful fuzzing

THANK YOU AND QUESTIONS!

UCI Samueli
School of Engineering
University of California, Irvine

清華大學
自強不息 厚德載物
TSINGHUA UNIVERSITY
1911