DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels

aka SAD DNS

Keyu Man, Zhiyun Qian, Zhongjie Wang,
Xiaofeng Zheng†, Youjun Huang†, Haixin Duan†
Contents

• Background
  • DNS Cache Poisoning
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
DNS Cache Poisoning

Trudy (Off-path)

Resolver

5.6.7.8

bank.com Nameserver (NS)

Alice’s Browser

Trudy

2.2.2.2

6.6.6.6

www.bank.com IP=??

www.bank.com IP=6.6.6.6

www.bank.com IP=6.6.6.6

Cached Wrong record!

www.bank.com IP=6.6.6.6

www.bank.com IP=6.6.6.6

www.bank.com IP=2.2.2.2

www.bank.com IP=6.6.6.6
DNS Cache Poisoning

Resolver

5.6.7.8
Trudy (Off-path)

www.bank.com IP=6.6.6.6

Our Side Channel:

Traditional:

Traditional: $2^{16} \times 2^{16} = 2^{32}$ (Impossible in short time)
Contents

• Background
• Part I: Infer Ephemeral Port
  • Ephemeral Port Type: public-facing vs. private facing
  • Method I: Direct Scan
  • Method II: Side-channel-based Scan
  • Measurements
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Port Inference: Basics

- **Attacker**
  - UDP dport=53
  - UDP dport=67
  - ICMP: 67 isn’t open

- **OS**
- **APP**
- **Resolver**

Packet

Listen on 53

unbound

Port Inference: Ephemeral Port Type

**RFC 8085**

UDP datagrams may be directly sent and received, without any connection setup. Using the sockets API, applications can receive packets from more than one IP source address on a single UDP socket. Some servers use this to exchange data with more than one remote host through a single UDP socket at the same time. Many applications need to ensure that they receive packets from a particular source address; these applications MUST implement corresponding checks at the application layer or explicitly request that the operating system filter the received packets.
Contents

• Background
• Part I: Infer Ephemeral Port
  • Ephemeral Port Type
  • Method I: Direct Scan
  • Method II: Side-channel-based Scan
  • Measurements
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Port Inference: Direct Scan

• Some DNS software call `bind()` or `sendto()`
  • Unbound, dnsmasq

• Hindrance: ICMP rate limit
  • Per-IP limit: 1 pps* on Linux

• Solution I: **IPv6** to bypass Per-IP limit

• Solution II: No IPv6? Request **IPv4** thru **DHCP**

• Solution III: Still doesn’t work? **Side-channel**-based port scan

*PPS=**P**ackets Per **S**econd
Contents

• Background
• Part I: Infer Ephemeral Port
  • Ephemeral Port Type
  • Method I: Direct Scan
  • Method II: Side-channel-based Scan
  • Measurements
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Port Inference: Private-facing Ports

Attacker

Resolver

Nameserver

DNS Query

(Anonymous Port) 1234->53

**UDP** dport=1234

**ICMP:** 1234 isn’t open

**UDP** dport=1234
Port Inference: IP Spoofing

Attacker 5.6.7.8

Resolver

Nameserver 5.6.7.8

---

UDP dport=1234

UDP dport=5678

ICMP: 5678 isn’t open
Port Inference:

- ICMP Global Rate Limit:
  - Limit sending rate
  - Shared by all IPs
Port Inference: How It Works

Resolver with **NO** port open

Nameserver

Attacker

---

Hit 50 closed ports

Counter=50

50 UDP Probes

50 ICMPs

Verification

Spoofed

Normal

---

Resolver with **ONE** port open

Nameserver

Attacker

Counter=50

Hit 49 closed ports & 1 open port

Counter=50-49=1

50 UDP Probes

49 ICMPs

Verification

ICMP Reply
Contents

• Background
• Part I: Infer Ephemeral Port
  • Ephemeral Port Type
  • Method I: Direct Scan
  • Method II: Side-channel-based Scan
  • Measurements
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Port Inference: Measurements

- Forwarders:

<table>
<thead>
<tr>
<th>Router</th>
<th>ICMP Reply</th>
<th>Global ICMP Rate Limit</th>
<th>Using connect ()</th>
<th>Allow Spoofing Public IP in LAN</th>
<th>Vulnerable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verizon Fios Gateway (G1100)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N/A</td>
<td>N</td>
</tr>
<tr>
<td>Xiaomi (R3)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y1</td>
</tr>
<tr>
<td>Huawei A1 (WS826)</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N/A</td>
<td>N</td>
</tr>
<tr>
<td>Netgear (WNDR3700v4)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y2</td>
</tr>
<tr>
<td>Arris Spectrum Gateway (TR4400)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y1</td>
</tr>
<tr>
<td>TP-Link (Archer C59)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y1</td>
</tr>
</tbody>
</table>

Y1: vulnerable to an insider attack. Y2: vulnerable to an attack requiring collaboration between an insider and outsider.
Port Inference: Measurement

- **Open Resolvers:**
  - **34%** Vulnerable

- **Well-known Public Resolvers:**
  - **12/14** Vulnerable

<table>
<thead>
<tr>
<th>Resolver</th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>8.8.8.8</td>
</tr>
<tr>
<td>Cloudflare</td>
<td>1.1.1.1</td>
</tr>
<tr>
<td>OpenDNS</td>
<td>208.67.222.222</td>
</tr>
<tr>
<td>Comodo</td>
<td>8.26.56.26</td>
</tr>
<tr>
<td>Dyn</td>
<td>216.146.35.35</td>
</tr>
<tr>
<td>Quad9</td>
<td>9.9.9.9</td>
</tr>
<tr>
<td>AdGuard</td>
<td>176.103.130.130</td>
</tr>
<tr>
<td>CleanBrowsing</td>
<td>185.228.168.168</td>
</tr>
<tr>
<td>Neustar</td>
<td>156.154.70.1</td>
</tr>
<tr>
<td>Yandex</td>
<td>77.88.8.1</td>
</tr>
<tr>
<td>Baidu DNS</td>
<td>180.76.76.76</td>
</tr>
<tr>
<td>114 DNS</td>
<td>114.114.114.114</td>
</tr>
<tr>
<td>Tencent DNS</td>
<td>119.29.29.29</td>
</tr>
<tr>
<td>Ali DNS</td>
<td>223.5.5.5</td>
</tr>
</tbody>
</table>
Contents

• Background
• Overview
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
  • Strategy I: Malicious Name Server
  • Strategy II: Response Rate Limiting
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Extend Attack Window

Client → Resolver → Attacker → Nameserver

Query

Fake Response

Response

Response
Extend Attack Window: Malicious Name Server

- Port open window: 0.1s -> 10s
- **Forwarder** attack only

```
```

<table>
<thead>
<tr>
<th><strong>Answer</strong></th>
<th><a href="http://www.atkr.com">www.atkr.com</a> CNAME</th>
<th><a href="http://www.victim.com">www.victim.com</a></th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.victim.com">www.victim.com</a> A</td>
<td>1.2.3.4</td>
<td></td>
</tr>
</tbody>
</table>
Contents

• Background
• Overview
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
  • Strategy I: Malicious Name Server
  • Strategy II: Response Rate Limiting
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Extend Attack Window: Against Resolver

RRL: 18% Deployed
Contents

• Background
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
• Our Attacks
  • Forwarder Attack
  • Resolver Attack
• Defenses
• Conclusion
• Disclosure
Forwarder Attack

• Strategy
  • Port inference: **DHCP** for 240 IPs & scan ports directly
  • Extend port open window: **malicious name server**

• Victim
  • Xiaomi R3
  • Upstream resolver: 1.1.1.1

• Attacker
  • Raspberry Pi, connected to Xiaomi via 2.4GHz Wi-Fi
Forwarder Attack: Results

• Success rate: 20/20

<table>
<thead>
<tr>
<th>Total time</th>
<th>DHCP time</th>
<th>Attack time</th>
<th>DHCP req’d</th>
<th>DHCP get</th>
<th>Port scanned</th>
</tr>
</thead>
<tbody>
<tr>
<td>268s</td>
<td>131s</td>
<td>137s</td>
<td>240</td>
<td>234</td>
<td>28383</td>
</tr>
</tbody>
</table>

Effective port scan speed: $28383 \div 137s = 207 \text{ pps}$
Contents

• Background
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
• Our Attacks
  • Forwarder Attack
  • Resolver Attack
• Defenses
• Conclusion
• Disclosure
Production Resolver Attack

```
$ dig @ test2.test.xiaofengtest.net +timeout=999

; <<< DIG 9.11.5-P4.5.1ubuntu2.1-Ubuntu <<< @ test2.test.xiaofengtest.net +timeout=999
; (1 server found)
; ; global options: +cmd
; ; Got answer:
; ; >>>HEADER<< opcode: QUERY, status: NOERROR, id: 7660
; ; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096

;; QUESTION SECTION:
test2.test.xiaofengtest.net. IN A

;; ANSWER SECTION:
test2.test.xiaofengtest.net. 300 IN A 1.2.3.4

;; AUTHORITY SECTION:
test2.test.xiaofengtest.net. 3534 IN NS ns.test2.test.xiaofengtest.net.

;; ADDITIONAL SECTION:
ns.test2.test.xiaofengtest.net. 294 IN A 54.177.157.64

;; Query time: 172 msec
;; SERVER: #53( )
;; WHEN: Thu Apr 02 20:54:05 UTC 2020
;; MSG SIZE rcvd: 105
```

20ms delay, 3ms jitter, 0.2% loss

Name Servers (Controlled by us)
## Resolver Attack: Results

<table>
<thead>
<tr>
<th>Attack</th>
<th># Back Server</th>
<th># NS</th>
<th>Jitter</th>
<th>Delay</th>
<th>Loss</th>
<th>Total Time</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tsinghua</td>
<td>2</td>
<td>2</td>
<td>3ms</td>
<td>20ms</td>
<td>0.2%</td>
<td>15 mins</td>
<td>5/5</td>
</tr>
</tbody>
</table>

Refer to the paper for more detailed results!
Contents

• Background
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Defenses

- DNSSEC
- 0x20 encoding
- DNS cookie
  - Only 5% open resolvers deployed
- Disable ICMP port unreachable
- Randomize ICMP global rate limit
Contents

• Background
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Conclusion

- Side-channel-based UDP port scan
- Make DNS cache poisoning possible again!
- Real-world attacks
Contents

• Background
• Part I: Infer Ephemeral Port
• Part II: Extend Attack Window
• Our Attacks
• Defenses
• Conclusion
• Disclosure
Disclosure
Thank you!

Zhiyun Qian, zhiyunq@cs.ucr.edu

https://github.com/seclab-ucr

@pkqzy888

SAD DNS website:
https://www.cs.ucr.edu/~zhiyunq/SADDNS.html
Practical Concerns

• Concerns on attacking resolver
• Cache Override
  • Inject non-existing NS record
• Multiple NSes
  • Flood all & spoof one to infer port and inject
  • NS pinning (valid on unbound)
• Multiple Backend Resolver
  • Attack them all together
  • Not too many

Infer Ephemeral Port II: Side Channel Scan

• Pinpoint to exact port #: Binary Search

<table>
<thead>
<tr>
<th>dport</th>
<th>50</th>
<th>51</th>
<th>52</th>
<th>53</th>
<th>54</th>
<th>55</th>
<th>56</th>
<th>57</th>
<th>58</th>
<th>59</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No ICMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No ICMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 1 1 1 1 1 1 55 56 57 1 1
1 1 1 1 1 1 1 55 56 57 1 1
1 1 1 1 1 1 1 55 56 1 1 1
1 1 1 1 1 1 1 57 1 1 1
1 1 1 1 1 1 1 1 57 1 1 1
Extend Attack Window: Measurement

• Alexa Top 100k Nameservers:
  • **18%** vulnerable (1k & 4k pps)
  • More with potential

Vulnerable Population in Alexa Top 100k

![Graph showing the vulnerable population in Alexa Top 100k nameservers.](image-url)