DNS Security

Prof. Dr. Haya Shulman

ATHENE | Goethe-Universität Frankfurt | Fraunhofer SIT

ECO Security Expert Talk

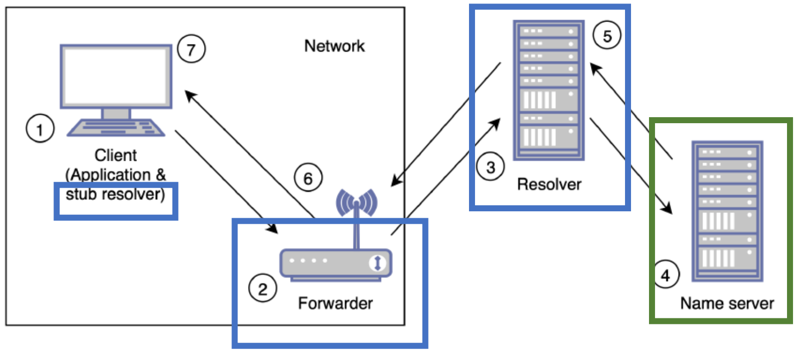
Videokonferenz, 22 Juni 2022

Overview

- DNS security in a nutshell
- TCP does not work
- New cache poisoning vector: injections over DNS
  - Validation of DNS inputs: who and where?
  - Injection attacks against applications and routers
- DNSSEC is vulnerable
- Conclusions

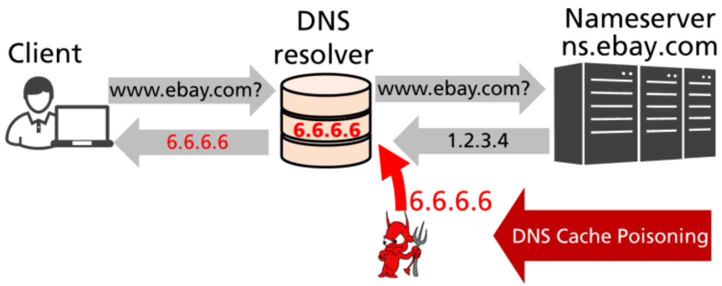
Domain Name System (DNS)

- Used to **lookup resources** and as a **platform for applications**
- **Resolvers** perform lookup for applications or users
  - Stub resolvers, forwarders, recursive resolvers
- **Nameservers** are hierarchical distributed database of resources



DNS cache poisoning

- Redirect victims to malicious hosts

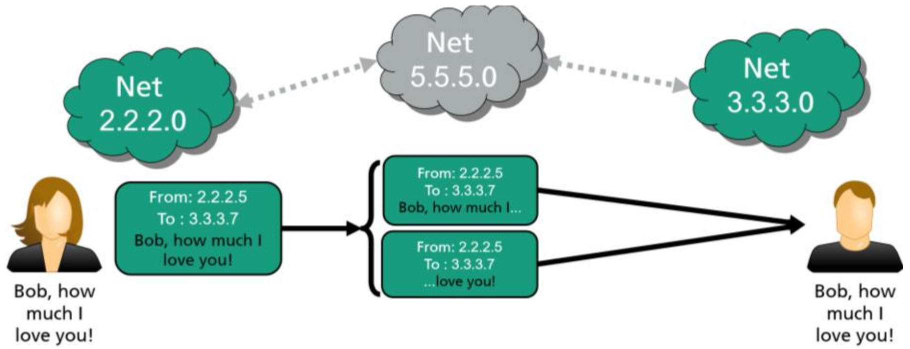


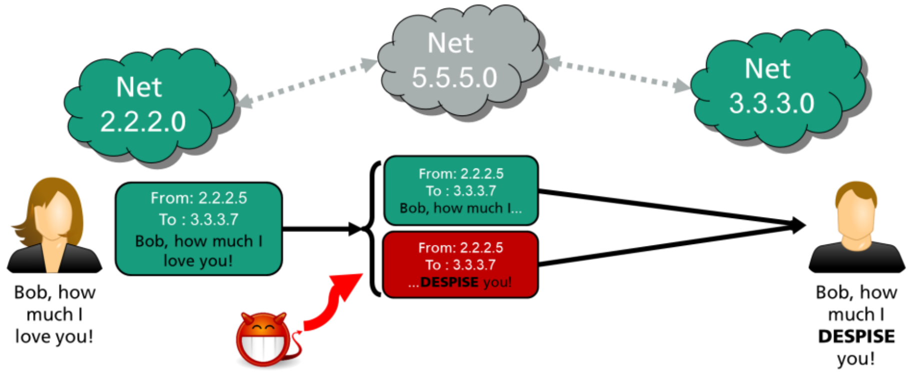
Bit	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31	
0	v4	IHL	TOS	Total Length	IP Header
32	IP Identifier		Flags	Fragment Offset	IP Header
64	Time To Live	Protocol	IP Header Checksum		IP Header
96	Source IP Address				IP Header
128	Destination IP Address				IP Header
160	Source Port		Destination Port		UDP Header
192	Length		UDP Checksum		UDP Header
224	Transaction Identifier (TXID)		DNS Flags		DNS Header
256	Question Count		Answer Record Count		DNS Header
288	Authority Record Count		Additional Record Count		DNS Header
…	Question Section				DNS Payload
	Answer Section				DNS Payload
	Authority Section				DNS Payload
	Additional Section				DNS Payload

- DNS request contains random values echoed in DNS response
  - Hijack BGP prefix to intercept DNS request
  - Side channels to hit the request values
  - Fragmentation to inject bogus content
- Successful cache poisoning attacks are challenging, require lots of work [CCS20, CCS21, Usenix21, Usenix22,…]

Recommended countermeasures:
→ DNSSEC validation of signed records against on-path
→ TCP against off-path

DNS cache poisoning via fragmentation





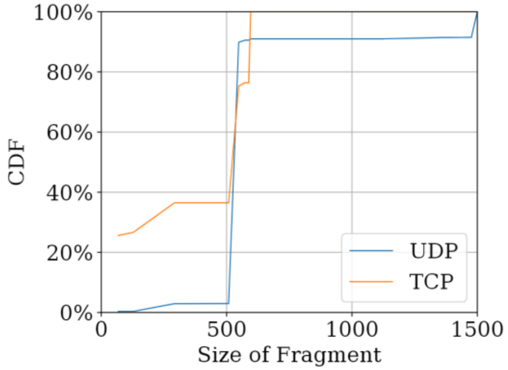
- Attack vector published in 2011 at IEEE CNS

DNS over TCP considered vulnerable

DNS responses are vulnerable to similar injection attacks like over UDP

- TCP fragmentation at the source: 393 additional vulnerable domains out of 100K Alexa

- TCP fragmentation at intermediate routers: > 600 routers in > 50 ASes



  - The fragments with TCP segments can be reduced to much lower sizes
  - Much more effective attacks

ANRW 2021
Applied Networking Research Workshop

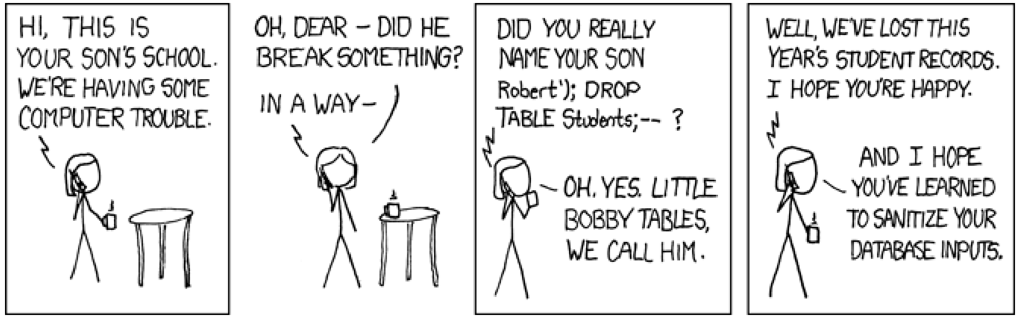
Overview

- DNS security in a nutshell
- TCP does not work
- New cache poisoning vector: injections over DNS
  - Validation of DNS inputs: who and where?
  - Injection attacks against applications and routers
- DNSSEC is vulnerable
- Conclusions

New attack vector: Injections over DNS

Well known: User inputs are not trusted!
Need to be sanitized/validated.

New: Attacks via inputs from other (trusted) sources

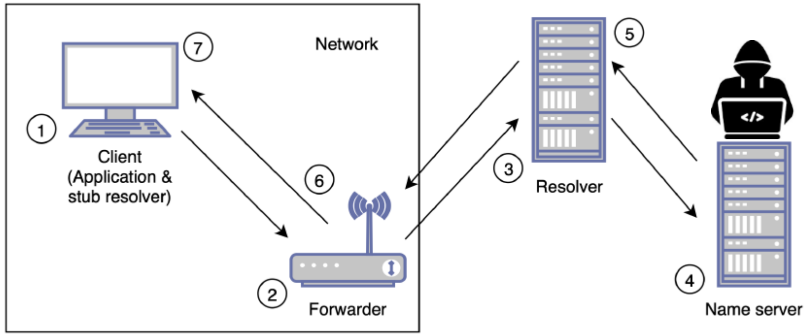


https://XKCD.com

“Be strict when sending and tolerant when receiving” [RFC1958]

- **DNS follows the end-to-end principle [RFC3597, RFC1035]**
  - Intermediate hosts (resolvers) should only interpret the data they need
  - Everything else forwarded unchanged
  - Allows easy adoption of new applications over DNS

- **We show DNS transparency can be exploited for injection attacks**



Components in DNS resolution chain

Application triggers a query…

1. **Nameserver provides records in line-format**
   - Record data can contain any value
   - Line format: List of labels, length of each label is prepended

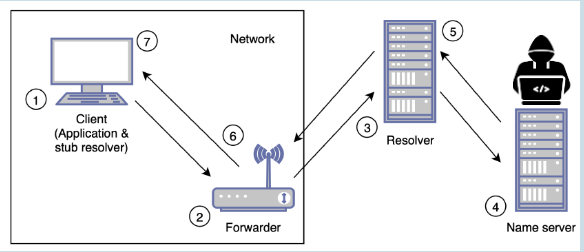
3. **Resolvers**
   - Treat DNS record data transparently

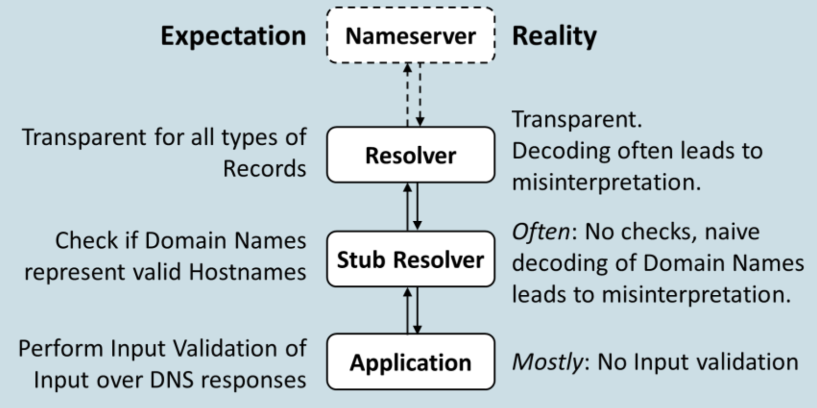
4. **Stub-resolvers / DNS-library**
   - Translates the line-format DNS data into textual form
   - Text format: Labels are separated with period (.)

5. **Application**
   - Uses the data

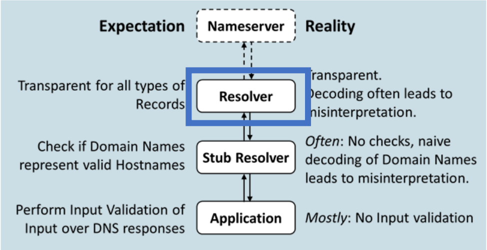
03	61	2e	62	02	3c	3e	03	63	6f	6d	00
label	a	.	b	label	<	>	label	c	o	m	label

Domain Name in **line-format**





Handling in DNS resolvers



- **DNS Resolvers handle DNS data transparently**
  - 96% of the tested resolvers (>1.3M) are standard compliant
- **What happens if**
  - Labels contain non-printable chars (i.e., NULL)
  - Labels contain periods (.) ?
- **Resolvers misinterpret** period-in-label, NULL
  - `www\.victim.com` → **`www.victim.com`**
  - `victim.com`**`\000.attacker.com`** → **`victim.com`**

Resolvers tested:

In lab:
7 recursive, 4 forwarders

Public:
11 public resolvers

In-the-wild:
1.3 million open resolvers from censys dataset

Cache poisoning via injection

- Trigger query for `attacker.com`, return `victim.com\000.attacker.com`
- Record in bailiwick: it is a subdomain of the domain in query `attacker.com`
- Record is processed and cached as `victim.com IN A` **6.6.6.6**

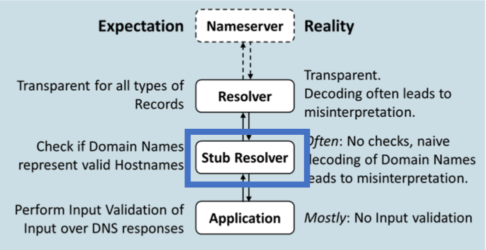
```
attacker.com                IN CNAME victim.com\000.attacker.com
victim.com\000.attacker.com IN A     6.6.6.6

victim.com                  IN A     1.1.1.1
```

100K open resolvers in the Internet vulnerable to cache-poisoning due to misinterpretation!

- **Cannot be prevented with DNSSEC**
- **Misinterpretation happens after DNSSEC validation**

Handling in stub resolvers



- **Domain names vs. hostnames [RFC952]**
  - Domain names can contain any data
    - Resolvers do not filter
  - Hostnames can only contain [a-z0-9-.]
  - POSIX specifies that libc resolver functions operate on hostnames not domain names
  - **Stub-resolvers should validate!**

- **But:**
  - Only **1 out of 10** validates
  - **7 out of 10** misinterpret zero or period

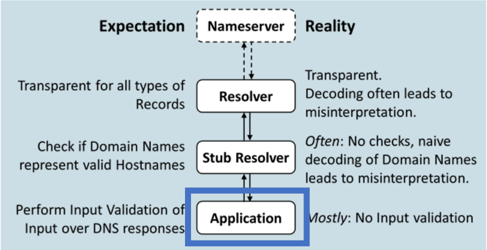
Test	Base	/	@	\.	\000	XSS	SQL	ANSI
Payload (Fig.9)	1.1.1.1	2.2.2.2	3.3.3.3	5.5.5.5	4.4.4.4	6.6.6.6	7.7.7.7	8.8.8.8
glibc	✓	✗	✗	✗	✗	✗	✗	✗
musl	✓	✓	✓	✓	✓	✓	✓	✓
dietlibc	✓	✓	✓	✓	✓	✓	✓	✓
uclibc	✓	✓	✓	✓	✓	✓	✓	✓
windows	✓	✓	✓	✓	✓	✓	✓	✓
netbsd	✓	✓	(✓)[2]	(✓)[2]	(✓)[2]	✓	✓	(✓)[2]
mac os x	✓	✓	✓	(✓)[2]	✓	✓	✓	✓
go*	✓	✓	✓	✓	(✓)[3]	✓	✓	✓
openjdk8*	✓	✗	✓	(✓)[2]	(✓)[3]	✓[4]	✓	✓
node	✓	✓	✓	(✓)[2]	✓	✓	✓	✓

✓: Vulnerable. [2]: output was escaped. [3]: Zero-byte did not stop output.
[4]: Alternative XSS payload with " " instead of "/".
∗: Uses system stub resolver by default but offers a builtin-one.

Stub resolver test results (PTR)

Handling in applications

Applications do not validate DNS records



- DNS data seems to come from the OS
  → developers tend to trust it
- Application developers are not DNS developers
  - Not aware that DNS records can contain any value
- Validation would be challenging to implement…
  detect decoding errors: `a\.b.com` or `a.b.com`
- **Vulnerabile to attacks**:
  XSS, Stack overflow, Buffer Overflow, Config injection, …

DNS Use-Case	Application	Trigger Query: Trigger	Trigger Query: Set	Uses libc	Vali-dates	Input use	Attack found
Address lookups (A, CNAME)	Chrome	js,html		yes	no	cache	no
	Firefox	js,html		yes	no	cache	no
	Opera	js,html		yes	no	cache	no
	Edge	js,html		yes	no	cache	no
	unscd	client app		yes	no	cache	no
	java	client app		both	no	cache	no
	ping(win32)	✗	✗	yes	no	display	yes
discovery (MX, SRV, NAPTR)	openjdk	login	✗	no	no	create URL	yes
	ldapsearch	login	✗	no	no	create URL	no
	radsecproxy	login		no	no	configure	yes
Reverse lookups (PTR)	ping(linux)	✗	✗	yes	no	display	yes
	trace(linux)	✗	✗	yes	no	display	yes
	OpenWRT	✗	ping	yes	no	display	yes
	openssh	login		yes	no	display,log	yes
Authentication (TXT, TLSA)	policyd-spf	SMTP		no	no	text protocol	no
	libspf2	SMTP		no	-	parse	yes
All	Resolvers	client app		no	some	cache	yes

Applications tested

None of the applications validate!

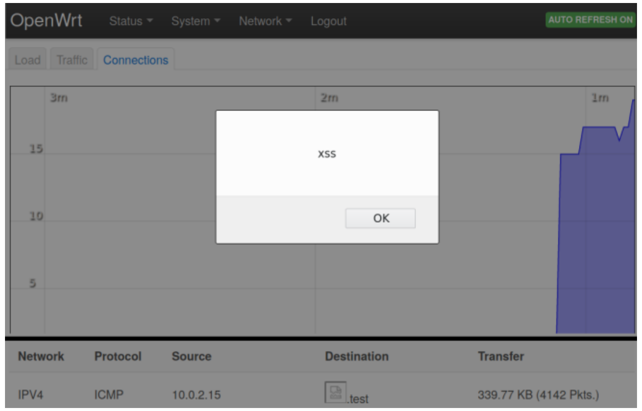
Injection attacks against applications

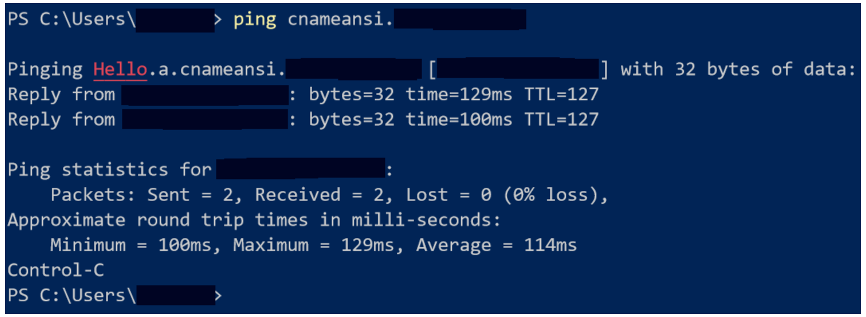
- Eduroam: international system for user identification in research institutions
- Vulnerability in Dynamic Peer Discovery of Eduroam
- The developers and DFN are notified
- CVEs registered and patches available

→ *Important: need to be installed manually*

Induced behaviour	Outcome
change dig DNS resolver	verification of vulnerability
pass `/some/file` as dig batch-file	disclose contents of `/some/file`
read `/dev/zero` as config file	100% CPU utilisation
provide malicious regex to regcomp()	radsecproxy crash
provide own RADIUS server and disable TLS-authentication	unauthorised network access

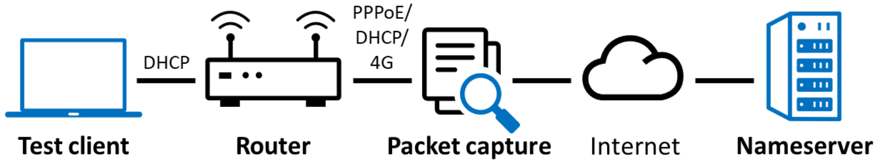
- XSS in OpenWRT
- ANSI escape code injection into ping





Injection attacks against residential routers

- Setup



- 15 (43%) routers vulnerable
  - 10 routers vulnerable to injections
  - 11 vulnerable to derandomization (TXID, ports)
  - 5 vulnerable to DNSSEC disabling
- 11 routers not standard compliant
  - E.g., no support of TCP

Vendor	Model	Has cache	version.bind	Attacks: Any attack	Misinterpretation injection: direct \.	direct \000	CNAME \.	CNAME \000	TXID forward	Fixed UDP port	CD=1 to disable DNSSEC	Non-standard
Home/SOHO routers												
Asus	GT-AC2900	✓	dnsmasq	-	-	-	-	-	-	-	-	-
AVM	Fritz!Box 6660	✓	-	✓	✓	-	-	-	-	-	-	-
AVM	Fritz!Box 7312	✓	-	✓	✓	-	-	-	-	-	-	-
AVM	Fritz!Box 7520	✓	-	✓	✓	-	-	-	-	-	-	-
AVM	Fritz!Box 7590	✓	-	✓	✓	-	-	-	-	-	-	-
Cudy	WR1300	✓	-[1]	-	-	-	-	-	-	-	-	-
D-Link	N150 (DIR-600)	✓	-[1]	-	-	-	-	-	-	-	-	-
D-Link	N300 (DWR-920)	✓	-[1]	-	-	-	-	-	-	-	-	-
DrayTek	Vigor2120	✓	-	✓	✓[5]	✓[5]	-	-	✓	-	✓	(h)
Edimax	N300	-	-	✓	-	-	-	-	-	✓(1027)	-	(h)
Linksys	E5350 (AC1000)	✓	dnsmasq-2.40	-	-	-	-	-	-	-	-	-
Linksys	EA8300 (AC2200)	✓	dnsmasq-2.78	-	-	-	-	-	-	-	-	-
Mercusys	MW305R	-	-	✓[2]	-	-	-	-	no[2]	-	-	(h)
Netgear	AC1200 / R6120	-	-	-	-	-	-	-	-	-	-	-
Netis	AC1200	✓	dnsmasq-2.79	-	-	-	-	-	-	-	-	-
STRONG	Wi-Fi Router 300	-	-	✓	-	-	-	-	-	✓(1027)	-	(h)
Tenda	AC10v3	✓	-	✓	✓	✓	-	-	-	✓(62066)	✓	(f),(h)
Tenda	F3	✓	-	✓	✓	✓	-	-	-	✓(50387)	✓	(f),(h)
TP-Link	Archer C7 (AC1750)	-	-	-	-	-	-	-	-	-	-	-
TP-Link	TL-WR841N	-	-	-	-	-	-	-	-	-	-	-
Trendnet	TW100-S4W1CA	✓	-	✓	-	-	-	-	-	✓(5530)	✓	(f),(h)
Xiaomi	MiRouter4A	✓	dnsmasq-2.71	-	-	-	-	-	-	-	-	-
Zyxel	Speedlink 5501	✓	dnsmasq-2.57	-	-	-	-	-	-	-	-	-
ISP-branded home routers												
Actiontec	MI424WR	-	-	✓	-	-	-	-	no[2]	✓(1024)	-	(h)
CenturyLink	C3000Z	✓	-	✓	✓	✓	-	-	✓	✓[3]	✓	(g),(h)
Telekom	Speedport Smart 3	✓	-[1]	-	-	-	-	-	-	-	-	-
Vodafone	Station TG3442DE	✓	dnsmasq-2.78	-	-	-	-	-	-	-	-	-
Small business routers												
Bintec	RS353a	✓	-	✓	-	-	✓	✓	-	-	-	(f),(h)
Cisco	RV260	✓	dnsmasq-2.78	-	-	-	-	-	-	-	-	-
Grandstream	GWN7000	✓	dnsmasq-2.78	-	-	-	-	-	-	-	-	-
Synology	RT2600AC	✓	dnsmasq-2.78	-	-	-	-	-	-	-	-	-
Ubiquiti	EdgeRouter4	✓	dnsmasq-2.78	-	-	-	-	-	-	-	-	-
Mobile/4G routers												
Huawei	5G CPE 5 Pro 2	✓	-	✓	-	✓[5]	-	-	-	-	-	(g)
Level421	TARKAN	✓	dnsmasq-2.51	-[4]	-[4]	-	-[4]	-	-	-	-	-
Teltonika	RUT950U022C0	✓	dnsmasq-2.81	-	-	-	-	-	-	-	-	-
SUM(✓)	35 100%	28 80%	- -	15 43%	8 23%	5 14%	1 3%	1 13%	4 11%	7 20%	5 14%	11 31%

✓: vulnerable/yes. -: not vulnerable/no. [1]: hidden dnsmasq version, [2]: uses sequential TXIDs, [3]: Port selected randomly at boot. [4]: ISP network vulnerable. [5]: Query section mismatch. (f): CNAME chain merging. (g): EDNS can cause broken responses. (h): No TCP support.

White-box analysis of firmware

- **Special character misinterpretation**
  - Vulnerable decoder implementations
  - Vulnerable cache implementations
    - `Qname-to-address map`
    - `Qname-to-packet map`
- **TXID forwarding:** forwarders extract min info from packets (TXID and qname) and ignore the rest
  - Do not change the TXID → forward as is
- **No source port randomization**
  - Some implementations set static port
  - Some chose with rand() C function: PRG is seeded with srand(time(NULL)) current UNIX timestamp in seconds from January 1, 1970
    - Should produce random time, but, the time is reset to January 1, 1970 after every reboot

Vendor	Model	OS	DNS forwarder implementation	Version	Open source	Vulnerabilities	Non-standard
Actiontech	MI424WR	linux	totd	1.5	✓	(c) (below 1.5.3), (d)	-
AVM	Fritz!Box 7590		"multid"		-	(a)	-
	Fritz!Box 6660						
	Fritz!Box 7312						
	Fritz!Box 7520						
Zyxel	C3000Z		dproxy-nexgen		✓	(a),(b),(d),(e)	(f),(g),(h)
Actiontec	V1000H[1]						
Huawei	5G CPE 5 Pro[1]		libmsgapi.so		-	(a)	(g)
	5G CPE 5 Pro 2						
Cudy	WR1300		dnsmasq	2.45	✓	-	-
Telekom	Speedport Smart 3			2.59			
D-Link	DIR-600			2.45			
	DWR-920			2.78			
Netgear	R6120		dnrd	2.19	✓	(a)	-
Tenda	AC10[1]			2.20.3		(a),(e)	
TP-Link	Archer C7		dnsproxy_deamon.sh		-	-	-
	TL-WR841N						
Strong	Wifi Router 300	eCos	(unnamed implementation)		-	(d)	(h)
Edimax	N300						
Tenda	F3		"DNS deamon"		-	(a),(d),(e)	(f),(h)
	AC10 v3						
Trendnet	TW100-S4W1CA		(unnamed implementation)		-	(d),(e)	(f),(h)
Mercusys	MW305R	VxWorks	dnsProxy.c		-	(c)	(h)
Bintec	RS353a	"BOSS"	"dnsd"		-	(a)	(h),(f)
DrayTek	Vigor2120	"DrayOS"	(unnamed implementation)		-	(a),(b),(e)	(h)

(a): Misinterpretation injection. (b): TXID forwarding. (c): Sequential TXID. (d): Fixed UDP port. (e): Disable DNSSEC via CD=1. (f): CNAME chain merging. (g): EDNS can cause broken responses. (h): No TCP support. [1]: Additional router not tested physically.

Vulnerable routers in the wild with ad network

- Create fingerprints of routers
  - Images routers use
  - Default addresses from factory settings
  - Special domain names used by vendors to redirect to web interface

- We embed a js on our website
  - We identified web interface in 973 clients
  - We found 929 vulnerable routers

% of Identified	% of Total	Absolute	Generic match	Vulne-rable	Router
41.62%	0.59%	405	x	x	Tenda
24.25%	0.34%	236	x	x	Huawei
15.42%	0.22%	150	x	x	Fritzbox
12.13%	0.17%	118	x	x	Mercusys
1.54%	0.02%	15	x		Linksys AC2200
1.34%	0.02%	13			Xiaomi Mi router
1.23%	0.02%	12		x	Draytek
1.13%	0.02%	11			Speedport Smart
0.72%	0.01%	7		x	Netgear R6120
0.21%	0.00%	2			D-Link DIR-600
0.10%	0.00%	1			Teltonika
0.10%	0.00%	1			Linksys AC1000
0.10%	0.00%	1		x	Centurylink
0.10%	0.00%	1			ASUS ROG
100.00%	1.42%	973	-	95.48%	Identified
-	98.58%	67482	-	-	Not Identified
-	100.00%	68455	-	1.36%	Total

Keep it Simple Stupid: also relevant for routers

- Many residential routers implement DNS forwarders
- But, do not implement most functionality and security features of DNS
- Remove DNS from routers: implement forwarding as a simple NAT rule
- Resolver of ISP eliminates performance penalty
  - The resolvers of the ISP are in proximity
  - The caches also include records cached on routers

Overview

- DNS security in a nutshell
- TCP does not work
- New cache poisoning vector: injections over DNS
  - Validation of DNS inputs: who and where?
  - Injection attacks against applications and routers
- DNSSEC is vulnerable
- Conclusions

DNSSEC is vulnerable and can be disabled

- Multiple algorithms were standardised for DNSSEC
  - Most zones are signed with RSA, some with ECDSA
  - Most resolvers support RSA and ECDSA

- Sign zones with new algorithms
  - Only unsupported algorithms → SERVFAIL or no validation
  - Supported and unsupported algorith→ in some cases leads to vulnerabilities (even with public DNS providers)

- Countermeasures can lead to failures, e.g., during key rollover

Conclusions

- Misinterpretations and wrong processing all the way
- Who should validate?
  - Applications do not know what should be the correct decoding
  - If DNS resolvers start validating:
    (1) Will lose transparency
    (2) Cannot know what is valid in advance
- Routers are mostly vulnerable
- Mitigations:
  - Resolvers: Test your resolver with https://xdi-attack.net/
  - Fix vulnerabilities: CVE-2021-20314, CVE-2021-32019, CVE-2021-2432, CVE-2021-32642, CVE-2021-33195, CVE-2021-3672, CVE-2021-22931, CVE-2021-43523,…
- Our works shows the complexity and challenges of content validation for zero trust security

Challenges with validation:
- Which inputs are illegal and should be filtered?
- What happens with new inputs that may be discovered?
- How to update all resolvers in the world to support this?

!תודה רבה

Merci beaucoup!

çok teşekkürler

谢谢

Thank you very much!

Dank je wel!

Vielen Dank!

Muchas gracias

ありがとうございます

Dziękuję!

zor spas

Grazie mille!

شكرا لك