DNS-over-TCP Considered Vulnerable

Tianxiang Dai
Fraunhofer SIT
Germany

Haya Shulman
Fraunhofer SIT
Germany

Michael Waidner
Fraunhofer SIT
TU Darmstadt
Germany

ABSTRACT

The research and operational communities believe that TCP provides protection against IP fragmentation attacks and recommend that servers avoid sending DNS responses over UDP but use TCP instead.

In this work we show that IP fragmentation attacks also apply to servers that communicate over TCP. Our measurements indicate that in the 100K-top Alexa domains there are 393 additional domains whose nameservers can be forced to (source) fragment IP packets that contain TCP segments. In contrast, responses from these domains cannot be forced to fragment when sent over UDP.

Our study not only shows that the recommendation to use TCP instead of UDP in order to avoid attacks that exploit fragmentation is risky, but it also unveils that the attack surface due to fragmentation is larger than was previously believed. We evaluate IP fragmentation-based DNS cache poisoning attacks against DNS responses over TCP.

CCS CONCEPTS

• **Security and privacy → Network security**.

KEYWORDS

DNS Cache Poisoning, IP Fragmentation, TCP

ACM Reference Format:
Tianxiang Dai, Haya Shulman, and Michael Waidner. 2021. DNS-over-TCP Considered Vulnerable. In *Applied Networking Research Workshop (ANRW '21), July 24–30, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3472305.3472884

1 INTRODUCTION

IP fragmentation allows to adjust packets to the size supported by the networks which the packets traverse. Given a too large packet, the source or the routers fragment packets into smaller fragments. The receiver reassembles the fragments back into the original IP packets. To identify the fragments that belong to the same IP packet the receiver uses a 16 bit IP identifier (IP ID) in the IP header during reassembly. The operating system at the sender assigns an IP ID to every outbound IP packet. If a packet is fragmented by the source or by the intermediate devices, each IP fragment receives the same IP ID value as the original IP packet. The receiving host identifies fragments with the same IP ID and reassembles them into original IP packet.

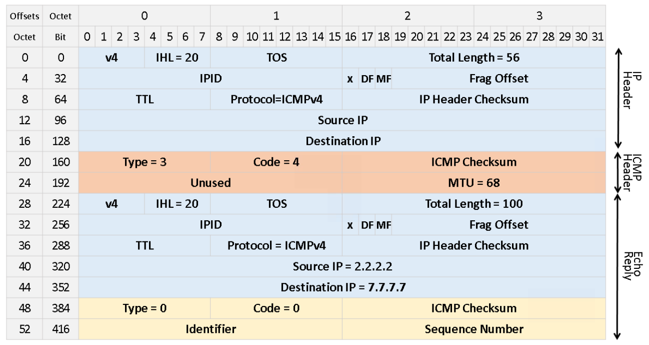
Computing the IP ID value. Some operating systems use a globally incremental counter to generate IP ID values: after a packet is sent, the IP ID counter is incremented and the next packet receives the next value. For instance, Windows (e.g., Windows 8 and 10), Android 5.0.1 and FreeBSD use a global counter [20]. Other operating systems, such as new Linux versions, use unpredictable IP ID assignment. For instance, Linux and MacOS to use local counters and OpenBSD use pseudo-random IP ID assignment [21]. In this work we focus on servers with global incremental IP ID counters.

Fragmentation-based attacks. The idea underlying the attacks that exploit fragmentation is to interfere in the IP reassembly process with fragment mis-association by sending a spoofed fragment to the victim client, which when reassembled with the genuine fragment from the server, results either in an incorrect fragment which is discarded by the client (causing DoS attack against the target service , such as [16]) or results in a correct fragment which contains malicious payload that was carried in the spoofed fragment [10]. Such attacks are typically launched against services that run over UDP, such as DNS, where the majority of the requests/responses are exchanged over UDP. Since most UDP traffic is not fragmented, the attackers can trigger source fragmentation by sending a spoofed ICMP fragmentation needed error message (type: 3, code: 4) to the target server. The ICMP packet should contain the original IP header and first 8 bytes of the payload that triggered the error message as well as the MTU of the router that sent the ICMP error, [RFC792] [22].

TCP uses path MTU discovery. To avoid fragmentation TCP performs path MTU discovery (PMTUD) and accordingly adjusts the Maximum Segment Size (MSS). To discover the MTU the sender transmits IP packets with a `do not`

ANRW '21, July 24–30, 2021, Virtual Event, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8618-0/21/07...$15.00
https://doi.org/10.1145/3472305.3472884

`fragment` (DF) bit set in the IP header. When a router with a smaller MTU receives such a packet it discards the packet and returns to the sender an ICMP fragmentation needed error message, illustrated in Figure 1. This causes the TCP at the sender to reduce the MSS to the value indicated in the MTU (minus 20 bytes) and resend the packet. The process iterates until the packet is received by the destination. As a result of PMTUD, services running over TCP are not subject to fragmentation and hence should not be vulnerable to fragmentation based attacks. Indeed, there are recommendations to use TCP when possible in order to avoid (DNS cache poisoning and DoS) fragmentation based attacks [2, 7, 18, 27]. Many DNS servers are configured to avoid fragmentation by lowering default EDNS buffer size and setting the TC bit on oversized DNS responses, signalling to the client that it should resend the DNS request over TCP.



Offsets Octet	Octet	0 (Bit 0–7)	1 (Bit 8–15)	2 (Bit 16–23)	3 (Bit 24–31)
Octet	**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	0	v4 IHL = 20	TOS	Total Length = 56	
4	32	IPID		x DF MF Frag Offset	
8	64	TTL	Protocol=ICMPv4	IP Header Checksum	
12	96	Source IP			
16	128	Destination IP			
20	160	Type = 3	Code = 4	ICMP Checksum	
24	192	Unused		MTU = 68	
28	224	v4 IHL = 20	TOS	Total Length = 100	
32	256	IPID		x DF MF Frag Offset	
36	288	TTL	Protocol = ICMPv4	IP Header Checksum	
40	320	Source IP = 2.2.2.2			
44	352	Destination IP = 7.7.7.7			
48	384	Type = 0	Code = 0	ICMP Checksum	
52	416	Identifier		Sequence Number	

Figure 1: ICMP Packet too big with `EchoReply`.

Fragmentation attacks against TCP. In this work we show that off-path attackers can force the servers to fragment communication over TCP. We perform a measurement study of the vulnerable servers. In contrast to common belief that TCP provides sufficient protection against off-path network adversaries we show that this is not so. We reveal that fragmentation based attacks against services over TCP allow to circumvent the entropy in TCP segments, including Sequence Number (SN) and source port. We explain the challenges that the network adversaries need to address to inject spoofed payload into TCP segments and demonstrate how forcing the servers to fragment traffic over TCP can be exploited for DoS and DNS cache poisoning attacks. In addition, fragmentation is also performed by the intermediate routers, when these have small next-hope MTU values. An analysis of CAIDA data-traces between 2008 and 2016 showed at least 1K intermediate routers in the Internet generate ICMP fragmentation needed packets with next MTU values even below 576 bytes, and fragment IP packets with TCP, despite path MTU discovery pf the source, [9].

Organisation. We review related work in Section 2. In Section 3 we provide measurements of servers that can be forced to fragment TCP traffic. In Section 4 we show measurements of IPID allocation in the Internet. In Section 5 we evaluate DNS cache poisoning attacks against DNS responses served over TCP. We conclude in Section 6.

2 RELATED WORK

IP fragmentation of UDP communication has a long history of attacks, including DoS (Denial of Service) attacks on IP defragmentation caches, [15, 16], and more recently IP fragments mis-association was exploited for injecting malicious content into NTP and DNS packets for shifting time and for launching DNS cache poisoning attacks respectively, [3, 10, 11, 13, 14, 19, 24, 25].

To reduce the threats introduced by fragmented IP packets that carry UDP datagrams best practices recommend to avoid fragmentation [1, 2, 5, 7, 18, 27]. Indeed, most servers were patched to avoid fragmentation, and use TCP for transmitting responses that are too large to fit in a single IP packet.

Is TCP really secure against IP fragmentation attacks? The possibility of fragmentation attacks against TCP was discussed in non academic community, starting with Zalewski who in 2004 in a bugtraq post[1] discussed a possibility for an attack that spoofs a non-first fragment of a TCP segment. There was however not a proof of concept of this attack and it was not validated in practice. In this work we demonstrate attacks that exploit IP fragmentation for injecting malicious records into DNS responses sent over TCP.

3 MEASURING TCP FRAGMENTATION

We perform a study of the Alexa Top-100K domains to infer the fraction of domains with nameservers that can be forced to fragment responses over TCP. For this, we first create a TCP connection to remote nameserver and send a DNS request over the connection. Once we receive the response, without acking, we send to the nameservers ICMP fragmentation needed packet, a.k.a. Packet Too Big (PTB, type: 3 and code: 4). Then we wait for retransmission and check if the response arrives fragmented, and record the minimum fragment size that we observe.

We evaluate reaction to three types of payloads embedded in ICMP PTB messages: TCP header, UDP header and ICMP echo reply. One example of ICMP PTB packet with ICMP echo reply as payload in showed in Figure 1. Considering that TCP is stateful, we also test two variations of TCP headers. One with wrong sequence number embedded in ICMP PTB (the TCP_Wrong row). The other with random port, thus for unexisting TCP connection (the TCP_New row). The results of our measurements are listed in Table 1.

[1]https://bugtraq.securityfocus.narkive.com/rm25I7e1/a-new-tcp-ip-blind-data-injection-technique

Out of 100K domains, we collected nameservers in 97,493 domains (column 'Total' in Table 1) by querying for NS record. 'N/A' indicates that we received no response. The valid responses are marked as 'Checked'. If we are able to cause nameservers in a domain to fragment responses, we mark the domain as 'Fragmented'. Column 'TC/NEW' has different meanings for UDP and TCP rows. For the UDP row, it says 13.77% domains have nameservers responding with TC (Truncated) bit set. For the TCP row, it shows the number of new domains that were identified as 'Fragmented', which do not fragment DNS responses over UDP. Our results show that 9,751 domains are vulnerable only to fragmentation attacks against UDP and 393 domains are vulnerable only to fragmentation attacks against TCP. More importantly, among the servers that are vulnerable to our IP fragmentation attacks over TCP, we find servers which actively avoid UDP by responding with a `TC` bit set and requesting that the DNS resolver re-sends the DNS request over TCP. Indeed, there are 76 of the 393 new domains which respond with `TC` bit but are vulnerable to our attacks. Namely, these servers implement the recommendations to move to TCP in order to avoid fragmentation based attacks to which communication over UDP is known to be vulnerable!

	Fragmented	Checked	N/A	TC/NEW	Total
UDP	10.11%	87.62%	12.38%	13.77%	100%
PTB_UDP	9,854	85,428	12,065	13,429	97,493
TCP	0.46%	90.30%	9.70%	0.37%	100%
PTB_TCP_Right	445	88,037	9,456	356	97,493
TCP	0.43%	89.42%	10.58%	0.34%	100%
PTB_TCP_Wrong	420	87,174	10,319	334	97,493
TCP	0.13%	89.51%	10.49%	0.07%	100%
PTB_TCP_New	130	87,263	10,230	72	97,493
TCP	0.15%	90.29%	9.71%	0.12%	100%
PTB_UDP	149	88,031	9,462	121	97,493
TCP	0.10%	89.99%	10.01%	0.10%	100%
PTB_EchoReply	95	87,732	9,761	94	97,493
TCP	0.51%	91.43%	8.57%	0.40%	100%
ALL	496	89,135	8,358	393	97,493

Table 1: Fragmentation of UDP and TCP in 100K-top Alexa domains.

To infer the fragment sizes that the vulnerable nameservers are willing to reduce to, we evaluated ICMP error messages with varying MTU sizes. We plot the distribution of the minimum fragment sizes in Figure 2. We find that domains supporting fragmentation over TCP allow much smaller fragments than UDP. About 40% of the domains that allow TCP fragmentation, can be reduced to MTU of 292 bytes or smaller, while 90% of the domains vulnerable to UDP fragmentation limit the minimum fragment size to at least 548 bytes.

We also measure the ratio of vulnerable nameservers within a domain. The results are plotted in Figure 3. Among

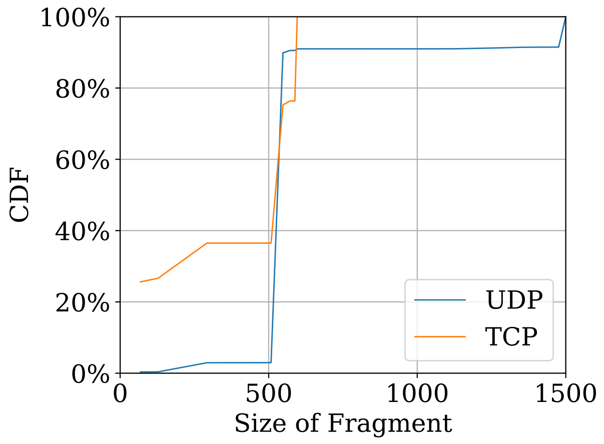


Figure 2: CDF of TCP and UDP fragment sizes.

the 496 domains with at least one nameserver vulnerable to fragmentation over TCP, 354 (71%) of them have more than half of their nameservers vulnerable. More specifically, all of the nameservers in 170 (34%) domains are vulnerable. Corresponding numbers for UDP are 9,065 (92%) and 7,706 (78%).

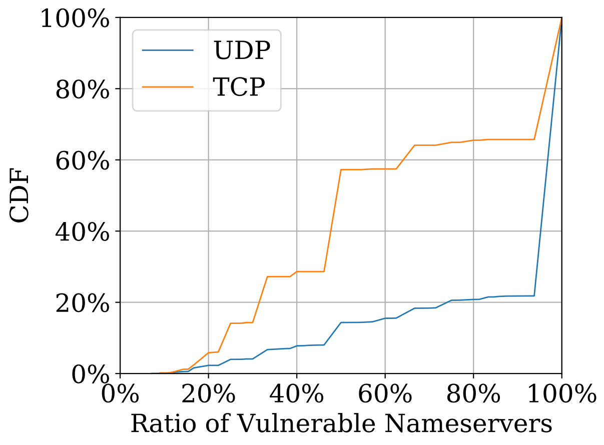


Figure 3: CDF of ratio of vulnerable nameservers for all vulnerable domains.

4 IP IDENTIFIER MEASUREMENTS

In this section we describe the IP ID allocation methods and report on the IP ID results we collected from the popular nameservers.

To identify the value of the IP ID we send packets from two hosts (with different IP addresses) to a nameserver.

IP Identifier. The 16 bit IP Identifier (IP ID) field in the IP header is used to identify fragments that belong to the same original IP packet [RFC791] [23]. The fragments are then reassembled by the recipient according to source and destination IP addresses, IP ID value and protocol field (e.g., TCP).

Global counter. Initially most operating systems used a globally incremental IP ID assignment which is easy to implement and has little requirement to keep state: just a single counter which is incremented with every packet that is sent. Global counters however were shown to be vulnerable to off-path attacks, [10]. A global counter is still popular in the Internet. Our study shows that 5.53% nameservers use global counter for UDP datagrams and 2.30% nameservers global counters for IP packets with TCP, see details in Table 2. To prevent the attacks some operating systems were patched to randomise their IP ID assignment.

In our work we focus on servers that implement globally incremental IP ID counters.

	Per-Host	Global	Zero	Random and other	N/A	Total
UDP	52.60%	5.53%	7.34%	33.40%	1.14%	100%
	51,281	5,388	7,152	32,560	1,112	97,493
TCP	14.43%	2.30%	75.92%	1.30%	6.04%	100%
	14,072	2,247	74,020	1,266	5,888	97,493

Table 2: IP ID allocation of in 100K-top Alexa.

5 OFF-PATH DNS POISONING OVER TCP

In this section we demonstrate cache poisoning attacks against DNS responses sent over TCP by injecting DNS records into fragmented IP packets. This shows that IP fragmentation attacks on TCP allow off-path attackers to bypass the randomisation with the sequence number and the source ports of TCP.

5.1 Attack Steps

In the first step of the attack, the adversary sends an ICMP echo reply message indicating a lower MTU. In the second step, the adversary finds out the IP ID value that will be assigned to the DNS response that the nameserver will send to the target victim. Next, the adversary crafts a spoofed second fragment with a TCP segment that contains a DNS record, and with the correct IP ID value, that matches the IP ID of the fragments sent by the nameserver. This spoofed fragment is stored in the IP defragmentation cache of the DNS resolver for 30 seconds. Note that the spoofed fragments are placed in advance. Thus there is no race condition. Subsequently, the adversary triggers a DNS request to the target domain. When the first genuine fragment of the DNS response arrives it is reassembled with the spoofed second fragment of the adversary and is moved on to TCP. If the checksum is correct, the ACK is in window, segment is accepted, and passed to DNS software. The ACK is in the window since it is the same sequence number (SN) value that was sent by the nameserver. Computing the TCP checksum is similar to computing the UDP checksum and is easy for an off-path attacker which knows the content of the original second fragment and knows which bytes it changed. Guessing the correct IP ID value depends on the rate at which the nameserver receives DNS requests. We explain next how to fix the TCP checksum and to extrapolate the IP ID value.

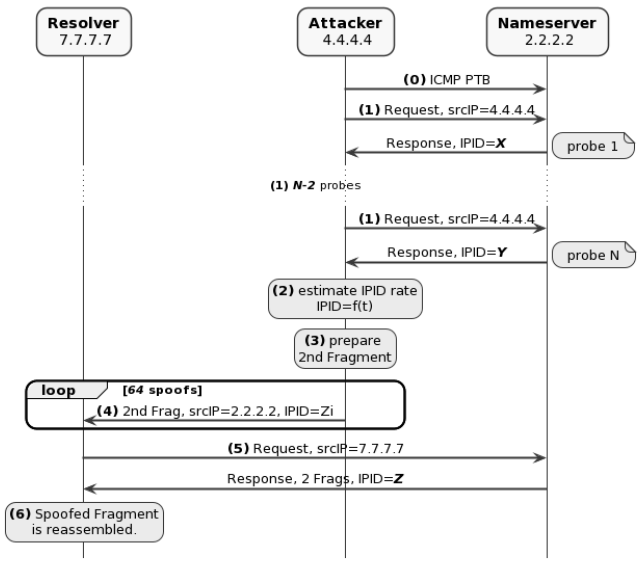

Resolver 7.7.7.7
Attacker 4.4.4.4
Nameserver 2.2.2.2
(0) ICMP PTB
(1) Request, srcIP=4.4.4.4
Response, IPID=X
probe 1
(1) N-2 probes
(1) Request, srcIP=4.4.4.4
Response, IPID=Y
probe N
(2) estimate IPID rate IPID=f(t)
(3) prepare 2nd Fragment
loop [64 spoofs]
(4) 2nd Frag, srcIP=2.2.2.2, IPID=Zi
(5) Request, srcIP=7.7.7.7
Response, 2 Frags, IPID=Z
(6) Spoofed Fragment is reassembled.

Figure 4: Traffic flow of the attack.

5.1.1 *Fixing the checksum.*

Since the spoofed fragment modifies parts of the payload of the genuine IP packet sent by the nameserver it also changes the checksum. Hence the adversary needs to adjust the checksum to ensure the checksum matches the one in the original IP packet, this is simple and done similarly to UDP [10].

5.1.2 *Predicting the IP ID value.*

We show our measurements of the IP ID algorithms in nameservers and then explain how we extrapolate the IP ID value in responses sent by busy nameservers.

IP ID measurements. Our measurements of the 100K-top Alexa domains show that 76,267 of the nameservers use predictable IP ID values in IP packets that contain TCP segments. In contrast, when communicating over UDP only 12,540 of the servers use predictable IP ID values in outbound packets.

Extrapolating IP ID. We first measure the rate at which the packets arrive at the nameserver. We do this by probing the nameserver's IP ID value at stable intervals. Let this rate be M packets per second. We next do an extrapolation of the IP ID value assuming that the server receives M packets per second. We use the following components: our prober and a nameserver that uses globally incremental IP ID. We use linear regression with Ordinary Least Square (OLS) method to estimate the relation between IP ID and timestamp t. Since IP ID is incremental, we assume:

$$IPID = a * t + b + \epsilon, \epsilon \sim N(0, \sigma^2)$$

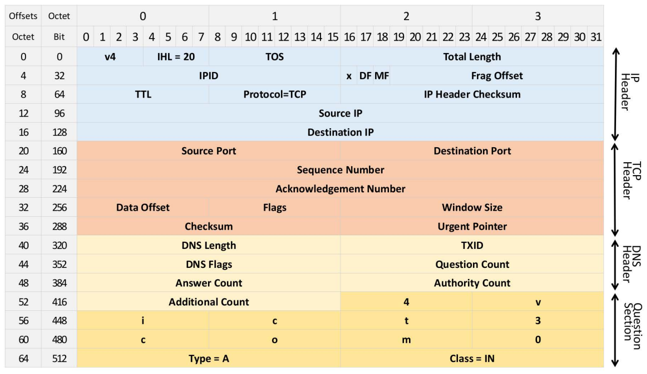
We send N probes to `2.2.2.2` (in step (1)). With N probes, we can estimate a, b and σ using OLS method in step (2).

We implement this extrapolation algorithm for probing the IP ID values prior to beginning of the attack.

5.2 Attack Evaluation

We set up an Unbound DNS resolver, and run the attack evaluations against the nameservers with global IP ID counters in 100K-top Alexa domains. Recent as well as more older approaches on IP ID prediction in old operating systems [6, 8, 17] demonstrate that our attacks have a much wider scope. We next show how to construct a second IP fragment with a malicious DNS record inside it, so that it is reassembled with the first fragment sent by the nameserver and results in a valid TCP segment.

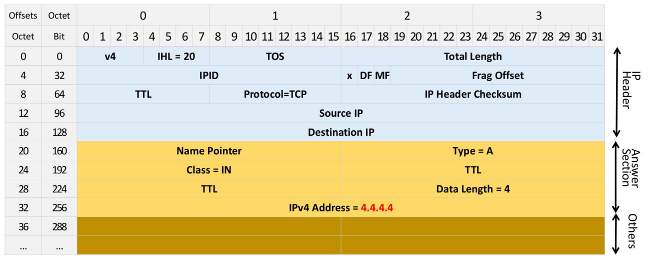
Figure 5 and Figure 6 are examples of two fragments of a DNS response over TCP. As we can see, all DNS related random challenges are in the first fragment, including TCP port and DNS Transaction ID (TXID). The only challenge the attacker needs to solve is to match the IPID. As explained in Section 5.1.2, it can be accomplished by probing and extrapolating. Afterwards, the attacker modifies the second fragment to include some malicious payload. To make the checksum pass, he can change few unused bytes in the packets, just as in [10]. Then he can spoof those second fragments to the resolver.



Offsets Octet	Octet Bit	0		1		2		3		
		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	0	v4	IHL = 20	TOS		Total Length				IP Header
4	32	IPID				x DF MF	Frag Offset			
8	64	TTL		Protocol=TCP		IP Header Checksum				
12	96	Source IP								
16	128	Destination IP								
20	160	Source Port				Destination Port				TCP Header
24	192	Sequence Number								
28	224	Acknowledgement Number								
32	256	Data Offset		Flags		Window Size				
36	288	Checksum				Urgent Pointer				
40	320	DNS Length				TXID				DNS Header
44	352	DNS Flags				Question Count				
48	384	Answer Count				Authority Count				
52	416	Additional Count				4		v		Question Section
56	448	i		c		t		3		
60	480	c		o		m		0		
64	512	Type = A				Class = IN				

Figure 5: First fragment, assuming MTU = 68

We evaluate the DNS cache poisoning attack that were previously shown to apply to UDP communication [10], against the servers that we found vulnerable to fragmentation attacks on TCP. The idea of the attacks is to inject a spoofed



Offsets Octet	Octet Bit	0		1		2		3		
		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	0	v4	IHL = 20	TOS		Total Length				IP Header
4	32	IPID				x DF MF	Frag Offset			
8	64	TTL		Protocol=TCP		IP Header Checksum				
12	96	Source IP								
16	128	Destination IP								
20	160	Name Pointer				Type = A				Answer Section
24	192	Class = IN				TTL				
28	224	TTL				Data Length = 4				
32	256	IPv4 Address = 4.4.4.4								
36	288									Others
...	...									

Figure 6: Spoofed second fragment, with DNS payload modified

fragment (with a source IP address of the victim nameserver) that contains a malicious payload. The spoofed fragment of the attacker is reassembled with the genuine first fragment from the nameserver. The resulting DNS response contains malicious records injected from the second spoofed fragment.

We simulate the success rate of the attacks using our DNS resolver with an IP defragmentation cache size of 64 packets, against nameservers with different IP ID increment rates Figure 7. The hitrate reaches 30% even at very high traffic rates of 1K packet per second.

The amount of packets sent during the whole attack is low: N probes to estimate server's IP ID rate and 64 spoofed second fragments, as showed in Fig. 4. Normally, fewer than 100 packets are required. Besides, the attacker does not need to repeat the probing phase for every attack. This makes the attack difficult to be detected.

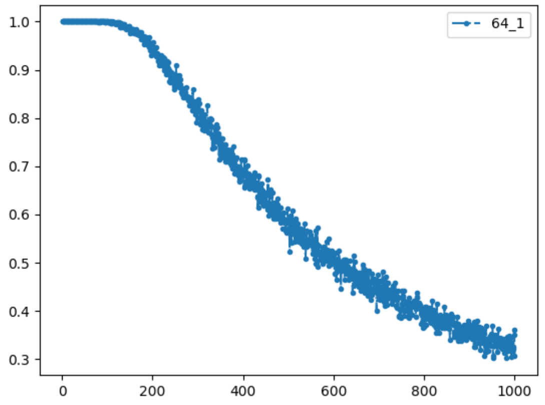


Figure 7: Attack hit rate for different traffic volumes. X-axis is the IP ID increment rate at nameserver (packet per second), Y-axis is success rate.

6 CONCLUSION: BUG OR FEATURE?

We show that the attacks which were believed to apply only to connectionless communication, such as UDP or tunnelling, also apply to IP communication with TCP. Our work shows that the recommendations to move to TCP to avoid the fragmentation attacks that apply to UDP, do not solve the problem.

The core issue is that the attacker can exploit ICMP messages with UDP datagrams as well as ICMP messages with echo reply packets to trigger fragmentation on TCP. Since these protocols allow fragmentation and do not trigger ICMP fragmentation needed error messages, the reaction of the operating systems to ICMP messages in such scenarios should be clarified in the standard. Our recommendation is that such ICMP messages are filtered in firewalls or at the servers.

To avoid DNS cache poisoning it is important to sign the zone files with DNSSEC [RFC4033-RFC4035]. Nevertheless, care should be taken as recent research demonstrated that some cache injections cannot be prevented even with DNSSEC, [12]. In addition, DNSSEC-signed domains can be vulnerable when the cryptographic material is weak or misconfigured, e.g., due to reuse of the shared modulus, [4, 26].

ACKNOWLEDGEMENTS

This work has been co-funded by the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) SFB 1119.

REFERENCES

[1] 2020. DNS Flag Day 2020. https://dnsflagday.net/2020/
[2] Ron Bonica, Fred Baker, Geoff Huston, Bob Hinden, Ole Troan, and Fernando Gont. 2019. *IP fragmentation considered fragile.* Technical Report. IETF Internet-Draft (draft-ietf-intarea-frag-fragile), work in progress ....
[3] Markus Brandt, Tianxiang Dai, Amit Klein, Haya Shulman, and Michael Waidner. 2018. Domain validation++ for mitm-resilient pki. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security.* 2060–2076.
[4] Taejoong Chung, Roland van Rijswijk-Deij, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M Maggs, Alan Mislove, and Christo Wilson. 2017. A longitudinal, end-to-end view of the {DNSSEC} ecosystem. In *26th {USENIX} Security Symposium ({USENIX} Security 17).* 1307–1322.
[5] J Dickinson, S Dickinson, R Bellis, A Mankin, and D Wessels. 2016. RFC7766: DNS transport over TCP-implementation requirements.
[6] Xuewei Feng, Chuanpu Fu, Qi Li, Kun Sun, and Ke Xu. 2020. Off-Path TCP Exploits of the Mixed IPID Assignment. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security.* 1323–1335.
[7] K Fujiwara and P Vixie. 2020. *Fragmentation Avoidance in DNS.* Technical Report. IETF Internet-Draft (draft-fujiwara-dnsop-avoid-fragmentation-03), work in progress ....
[8] Yossi Gilad and Amir Herzberg. 2011. Fragmentation considered vulnerable: blindly intercepting and discarding fragments. In *Proceedings of the 5th USENIX conference on Offensive technologies.* 2–2.
[9] Matthias Göhring, Haya Shulman, and Michael Waidner. 2018. Path MTU Discovery Considered Harmful. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS).* IEEE, 866–874.
[10] Amir Herzberg and Haya Shulman. 2013. Fragmentation Considered Poisonous, or: One-domain-to-rule-them-all.org. In *2013 IEEE Conference on Communications and Network Security (CNS).* IEEE, 224–232.
[11] Amir Herzberg and Haya Shulman. 2013. Vulnerable delegation of DNS resolution. In *European Symposium on Research in Computer Security.* Springer, 219–236.
[12] Philipp Jeitner and Haya Shulman. 2021. Injection Attacks Reloaded: Tunnelling Malicious Payloads over {DNS}. In *30th {USENIX} Security Symposium ({USENIX} Security 21).*
[13] Philipp Jeitner, Haya Shulman, and Michael Waidner. 2020. The Impact of DNS Insecurity on Time. In *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN).* IEEE, 266–277.
[14] Philipp Jeitner, Haya Shulman, and Michael Waidner. 2020. Pitfalls of Provably Secure Systems in Internet The Case of Chronos-NTP. In *2020 50th Annual IEEE-IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S).* IEEE, 49–50.
[15] Charlie Kaufman, Radia Perlman, and Bill Sommerfeld. 2003. DoS protection for UDP-based protocols. In *Proceedings of the 10th ACM conference on Computer and communications security.* 2–7.
[16] Christopher A Kent and Jeffrey C Mogul. 1987. *Fragmentation considered harmful.* Vol. 17.
[17] Amit Klein and Benny Pinkas. 2019. From IP ID to Device ID and KASLR Bypass. In *28th {USENIX} Security Symposium ({USENIX} Security 19).* 1063–1080.
[18] LetsEncrypt. 2018. Mitigating DNS Fragmentation Attack. https://community.letsencrypt.org/t/mitigating-dns-fragmentation-attack/74838.
[19] Aanchal Malhotra, Isaac E Cohen, Erik Brakke, and Sharon Goldberg. 2016. Attacking the Network Time Protocol.. In *NDSS.*
[20] Sophon Mongkolluksamee, Kensuke Fukuda, and Panita Pongpaibool. 2012. Counting NATted hosts by observing TCP/IP field behaviors. In *2012 IEEE International Conference on Communications (ICC).* IEEE, 1265–1270.
[21] Liran Orevi, Amir Herzberg, and Haim Zlatokrilov. 2018. DNS-DNS: DNS-based de-nat scheme. In *International Conference on Cryptology and Network Security.* Springer, 69–88.
[22] Jon Postel. 1981. Internet Control Message Protocol darpa internet program protocol specification. *RFC 792* (1981).
[23] Jon Postel. 1981. Internet protocol—DARPA internet program protocol specification, rfc 791. (1981).
[24] Haya Shulman and Michael Waidner. 2014. Fragmentation considered leaking: port inference for dns poisoning. In *International Conference on Applied Cryptography and Network Security.* Springer, 531–548.
[25] Haya Shulman and Michael Waidner. 2015. Towards security of internet naming infrastructure. In *European Symposium on Research in Computer Security.* Springer, 3–22.
[26] Haya Shulman and Michael Waidner. 2017. One key to sign them all considered vulnerable: Evaluation of {DNSSEC} in the internet. In *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17).* 131–144.
[27] Liang Zhu, Zi Hu, John Heidemann, Duane Wessels, Allison Mankin, and Nikita Somaiya. 2015. Connection-oriented DNS to improve privacy and security. In *2015 IEEE symposium on security and privacy.* IEEE, 171–186.