A Comprehensive Study of DNS-over-HTTPS Downgrade Attack

Qing Huang
University of California, Irvine
qhuang6@uci.edu

Deliang Chang*
Tsinghua University
changdl16@mails.tsinghua.edu.cn

Zhou Li
University of California, Irvine
zhou.li@uci.edu

Abstract

DNS-over-HTTPS (DoH) is one major effort to protect DNS confidentiality and integrity, which has been deployed by most of the popular browsers. However, we found this effort could be tainted by the downgrade attack, which exposes the content of DNS communications to attackers like censors. Specifically, we examined 6 browsers with 4 attack vectors that are relevant to our attack model and found all combinations that lead to successful attacks. The fundamental reason is that all browsers enable Opportunistic Privacy profile by default, which allows DoH fall backs to DNS when DoH is not usable. However, it is still concerning that none of the browsers attempt to notify users when such a change happens and some browsers take a long time to recover to DoH. At the end of the paper, we propose some countermeasures and we call for discussions from the Internet community to revisit the standards and implementations about DoH and usage profiles.

1 Introduction

Domain name system (DNS) maps human-friendly string-type domain names to machine-friendly numerical IP addresses, and it is a critical infrastructure of the Internet. Since the beginning of the Internet, DNS queries and responses are transmitted in plaintext according to RFC 1035 [29], which makes it very easy to be eavesdropped and manipulated. As a result, DNS is constantly under attack by network adversaries for surveillance and censorship [3, 18, 26].

To mitigate these threats and protect DNS's authenticity, confidentiality and integrity, several protocols aiming to transmit DNS packets in encrypted channel are proposed [14, 21, 34]. Among those methods, DNS-over-HTTPS (DoH) [34] is most promising, since it has already been implemented and integrated into most of the popular browsers like Google Chrome [17] and Firefox [30]. It is also offered as a service by large public resolvers like Cloudflare [12]. In essence, DoH transmits DNS queries and responses between the stub resolvers and recursive resolvers via HTTPS protocol, as such any applications supporting HTTPS can issue DoH queries. Comparing to DNS-over-TLS (DoT) [21] which requires a specialized stub resolver, the deployment overhead of DoH is much lower on the client-side.

To circumvent the protection offered by DoH, an active adversary might try to *downgrade* DoH to DNS and carry out the known DNS attacks. In fact, this is feasible as the usage profile RFC [35] specifies that a stub resolver can choose Opportunistic Privacy profile, which allows DNS encryption to fall back to plaintext when the encrypted channel cannot be constructed. Still, there exists a gap between RFC and the implementations done by browser vendors, and this leads to several questions we are interested in: 1) What kind of attack vectors would cause the DoH to be downgraded? 2) What is the reaction of a browser under the downgrade attack? 3) What could be improved when facing downgrade attack?

In this study, we start by revisiting the process of DoH resolution and identify the attack surface that can be exploited for DoH downgrade. Then we tested 4 attack vectors, including DNS traffic interception, DNS cache poisoning, TCP traffic interception and TCP reset injection on 6 browsers supporting DoH. We have reported our findings to the browser vendors. Though none of the browser vendors replying to us consider that browsers vulnerable to DoH downgrade attacks as security bugs, we argue that their design could be improved, as a user is never notified when DoH fallback happens, and some browsers have very long recovery time. Our contributions are listed below:

- We perform the *first* study of downgrade attacks on DoH, by systematically enumerating the attack surface and examining the attack vectors.
- We evaluate the attacks in a realistic lab environment and found downgrade attack is not only feasible but succeeding against all browsers. We also found the reactions of browsers under attack are concerning.

*Partially sponsored by China Scholarship Council. Work was done when visiting University of California, Irvine.

- We discuss the possible countermeasures at the implementation and protocol level.

2 Background

Domain name system (DNS). DNS translates domain names to their corresponding IP addresses. To be more specific, a software on client called stub resolver collects domain names requested by user applications. Then it sends DNS queries to recursive resolvers (RR). RR works as agent in DNS resolution. If the queried domain is not in its cache, it will send queries to authoritative name servers recursively. After RR receives the answer, it will send the answer back to the clients. So far, most of the DNS queries and responses are transmitted in plaintext, making it vulnerable to be manipulated. The DNS packets between stub resolver and recursive resolver are the major target for attackers, and previous research have shown that DNS queries from users could be used to track [19, 24] or censor [3, 26] them.

DNS-over-HTTPS (DoH). To mitigate the privacy issues of DNS, DoH is proposed to protect the connection between end-users and recursive resolvers. It uses HTTPS to encrypt DNS queries. DoH runs on TCP port 443, just like normal HTTPS. DNS requests are sent in the format of an URI template (e.g., `https://dns.google/dns-query{?dns}` is for Google public DNS). The domain name in the URI is used not only to find IP address of DoH resolver (through plaintext DNS resolution), but also to verify its identity (through SSL certificate verification). DoH is often provided by browsers as an integrated module. As such, DoH communications are opaque to operating systems.

3 DoH Downgrade Attacks

As TLS of HTTPS provides strong privacy and security protection, for the existing DNS attacks to succeed, a natural idea for the adversary is to *downgrade* DoH communication back to plaintext DNS. This is possible as the browser might try to achieve incremental deployment of DoH and avoid the disruption to users' normal communications when DoH is not usable. Below we first review the entire resolution process of DoH. After that, we describe the adversary model and available attack vectors that can be leveraged for DoH downgrade attacks.

3.1 Process of DoH Communication

Based on our analysis on popular browsers, DoH communication usually follows two phases as illustrated in Figure 1.

Phase 1: URI Resolution. URI used in HTTP requests of DoH is defined by a URI template [34]. Before DoH communication, a browser sends an *unencrypted DNS* request to resolve the URI and obtain the IP address of the DoH server (e.g., Google Public DNS and Cloudflare DNS). This phase is the same as the traditional DNS resolution process, which means any attacker capable of sniffing network traffic can view the plain text content in the DNS packet and tamper it.

Phase 2: Connection & Communication. The browser establishes a secure connection with the DoH resolver via TLS. After the connection is established, the DNS request will be encapsulated in an encrypted HTTPS packet through this transmission channel. The browser sends wire-format DNS messages under HTTPS GET or POST requests. An attacker able to compromise this phase could force the browser to fall back to plaintext DNS.

Additionally, for browsers such as Chrome, a mapping table is used to convert the DNS resolver configured in the operating system into its equivalent DoH resolver URI before Phase 1. We call it *Phase 0*. Phase 0 is often hard-coded in the browser software, thus we do not consider this phase to be attacked.

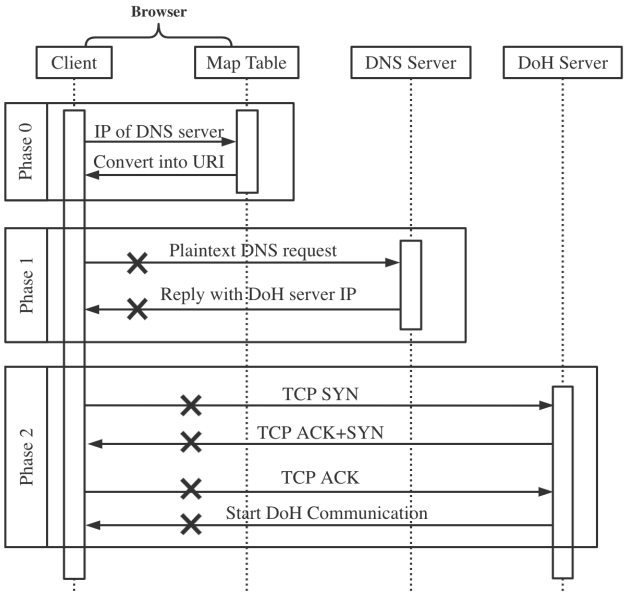

Figure 1: DoH resolution process. Crosses represent the attack surface.

3.2 Adversary Model

The attacker's goal is to force the encrypted DoH falling back to plaintext. In this paper, we assume two types of adversaries based on their capabilities of manipulating network packets.

In-path Attackers. They can inspect the traffic of the victim, and have the ability to modify all packets from and towards the victim. A good example of in-path attackers is a network gateway, which is usually controlled by the network administrators of a company or the owner of a public WiFi. As such those parties have the capability to perform in-path attack.

Another example is an adversary in the local network. The attacker could perform ARP cache poisoning attack [32] to redirect the victim traffic to the attacker's machine and act as a rogue middlebox.

On-path Attackers. They can inspect the traffic of the victim and inject new packets. But unlike in-path attackers, they are unable to intercept or modify the passing-through packets. On-path attacker is weaker than in-path attackers. Because it consumes fewer resources for attack, it is widely deployed in middleboxes doing censorship [26], resulting in attacks that scale at ISP or country level. On the other hand, an attacker close to the victim, who shares the same LAN can also become a on-path attacker. Sniffing the traffic associated with the victim host is necessary but there are various ways to achieve this prerequisite, regardless of whether the network communication is encrypted or not. For example, for wireless communication encrypted under WPA/WPA2, if the EAPOL frame of the four-way handshake can be obtained at the beginning of the connection between the victim host and the AP (Access Point), the encryption provided by some WiFi devices can be breached [38]. To hide the attacker's traits, the IP address can be spoofed with the original one in communication by the victim.

Putting the two attacks together, both attackers should have the ability to sniff the DoH traffic from client-side. The in-path attackers need the ability to intercept the packets of the victim, but the on-path attackers only need the ability to inject new packets.

3.3 Attack Method

Based on the workflow of DoH described in Section 3.1, we use the following attacking vectors to achieve downgrade attacks. Here we present four concrete of attack methods that can be leveraged by in-path and on-path attackers, targeting different phases of DoH.

DoH Server	Domain name
Google	dns.google
Cloudflare	chrome.cloudflare-dns.com[1] cloudflare-dns.com
Quad9	dns.quad9.net
Umbrella/OpenDNS	doh.opendns.com
CleanBrowsing	doh.cleanbrowsing.org
Comcast	doh.xfinity.com
DNS.SB	doh.dns.sb

Table 1: Domain names of to DoH resolvers.

DNS Traffic Interception. In-path attackers target Phase 1. If an attacker has the ability to modify the network packets passing through his/her own device, then he/she can attack the URI resolution phase of DoH by simply blocking the specific DNS traffic sent by the victim to obtain the IP address of DoH server. The specific DNS traffic can be filtered by the URI in DNS request. We have listed some DoH resolvers and their corresponding domain names used in resolve phase in Table 1.

DNS Cache Poisoning. On-path attackers target Phase 1. DNS cache poisoning refers to spoofing the DNS cache by sending back a response DNS packet to the victim with a fake or unreachable IP address instead of IP address of the target DoH server. In this case, the connection request will be redirected to the fake or unreachable IP address. Since browser cannot establish a connection with the correct DoH server, it will theoretically fall back to plaintext DNS transmission.

TCP Traffic Interception. In-path attacker targets Phase 2. Similar to DNS traffic intercepting, instead of attacking phase 1, this method tries to block the TCP traffic from phase 2 to force the DoH fallback to plaintext DNS. This can only be used by an in-path attacker who is able to modify the network packets.

TCP Reset Injection. More subtly, tampering the TCP traffic from connection & communication phase can be achieved by an on-path attacker. The attacker sniffs the network traffic exchanged between the victim and DoH resolver. Then the attacker obtains the sequence number and acknowledgment number in TCP headers, and send forged TCP reset packets to the victim and/or the DoH resolver to trick them to cut off the TCP connection. Similar to DNS cache poisoning, this attack method does not require the ability to intercept or modify existing packets.

Among those methods, DNS poisoning and TCP RST injection are known to be used in large-scale censorship [11,26]. Assume a censor adds domain names listed in Table 1 to its censoring list, and use DNS poisoning to forge the response to an unreachable IP address. The user of those DoH server will be forced to use plaintext DNS thus are subject to further censorship and surveillance. This example shows that censors could easily achieve censorship on DoH with their existing facilities.

4 Attack Evaluation

We implement our downgrade attack methods on DoH provided by different resolvers and browsers under different platforms. In this evaluation, we measured the behavior of various browsers that support DoH after being attacked, through network traffic analysis.

4.1 Experiment Setup

Evaluation Settings. All of our evaluation tasks focus on the reaction of browser when they face different attack methods. We selected 6 browsers listed in Table 2, which are all popular

[1] Only for Chrome browser.

Browser	Config	Profile	BType	Notif
Chrome 84.0.4147.89	OS&URI	Opportunistic*	Chrome+	No
Firefox 76.0.1	URI	Opportunistic*	Firefox	No
Edge 84.0.522.40	OS	Opportunistic	Chrome+	No
Brave 1.11.97	OS	Opportunistic	Chrome+	No
Opera 69.0.3686.77	URI	Opportunistic	Chrome+	No
Vivaldi 3.1.1929.458	OS	Opportunistic	Chrome+	No

Table 2: Browser DoH settings. "Config" indicates how to set DoH resolver's URI. "Profile" is the type of usage profile [35]. "BType" categorizes how a browser reacts when facing DoH Downgrade Attack. "Notif" is whether browser notifies its user when DoH is not usable.

and supporting DoH. We set up the testing environment with three machines (Windows laptop and MAC laptop as victims and one Debian Linux machine as attacker) connecting to a wireless router (AT&T WiFi Gateways). The attacker uses Wireshark 3.0.3 [33] to eavesdrop victim's traffic and Scapy-2.4.3 [6] to craft attack packets. For in-path attack scenario, we change the router's firewall to block TCP/DNS packets related to DoH, or redirect user's traffic to attacker's machine by ARP spoofing and intercept victim's packets on it. We let the victim configure a DoH server first and then visit several random websites. Both Phase 1 (URI Resolution) and Phase 2 (DoH Connection & Communication) are tested. Though we examined different DoH servers listed in Table 1, we found whether the attack succeeds or not is independent of this factor. Therefore, we focus on the browser side for the remaining evaluation.

Browser DoH Settings. Table 2 lists the detailed DoH settings of each browser. Chrome, Firefox and Opera allow a user to specify DoH resolver's URI in security settings panel, while all others carry out Phase 0 to obtain the URI. Chrome uses DNS provider configured in operating system by default and all other browsers (Edge, Brave, Vivaldi) only use DNS provider configured in operating system as their DoH provider [17, 30]. More importantly, we found how browser reacts to downgrade attacks depends on the *usage profiles* that are enabled by default or by the user. Similar to DoT usage profiles described in RFC 8310 [35], there are two options: *Strict Privacy profile* and *Opportunistic Privacy profile*. For the first option, when DoH communications cannot be established, e.g., that the resolver cannot be connected through DoH, a "hard fail" will happen such that client will not consider plaintext DNS as the backup plan. For the second option, the client will attempt to establish the connection with the DNS resolver and use plaintext DNS after DoH communication fails. Apparently, when Opportunistic Privacy profile is enabled, downgrade attacks have potential to succeed. Interestingly, we found *all* browsers enable the latter one *by default*. Switching to strict mode is feasible, *but not on every browser*. For example, Firefox enables the strict mode when a user visits `about:config` page and change `network.trr.mode` to `3`. Chrome on Windows can be switched to the strict mode by explicitly selecting or inputting a custom provider. For other browsers like Brave and Chrome on MAC, we have not discovered the interface. Also, there is a specific option in Firefox relevant to our attack: if the `network.trr.bootstrap_address` and `network.trr.URI` fields are set to the same DoH provider, Firefox can bypass the URI resolution phase and directly establish a DoH secure connection with the bootstrap address. As such, the attack methods against Phase 1 are ineffective in this case.

4.2 Browser Reaction under Attack

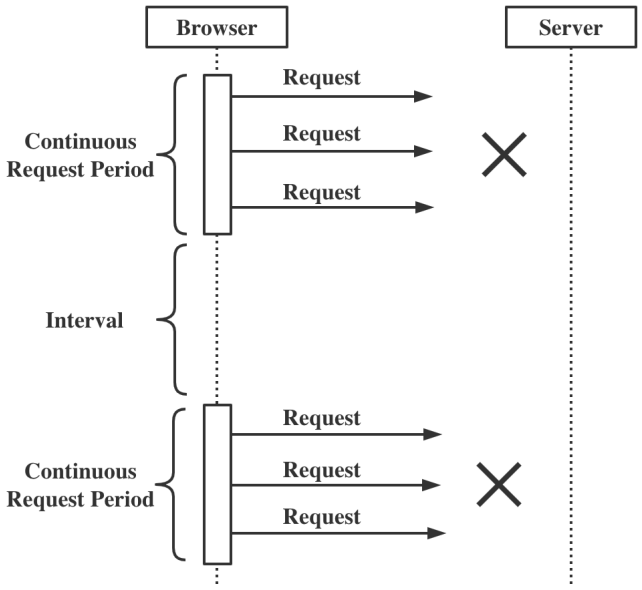

Figure 2: Reconnection Process after Failed.

We evaluated the reactions of the 6 browsers when they are facing 4 different attack vectors, using the default settings (Opportunistic Privacy profile). We found that no matter what attack vector this browser is facing, as long as the attack has hindered the network traffic of the DoH service, the browser's response behavior will follow a pattern, which can be described by three attributes: Continuous Request Period (CRP), Interval Growth (IG) and Max Interval (MI). These three attributes reflect how browser reconnects to DoH servers when under attack and we illustrate this process at a high level in Figure 2. Due to the high complexity of browser codebase, we decide to treat them as blackbox, and run the testing scripts dynamically and log the values we observe for the three attributes. Below we explain them and highlight the interesting observations.

Continuous Request Period (CRP). When the browser discovers that the connection fails, it will keep trying to send

multiple reconnect requests within some period. We define this period as CRP. From an attacker's perspective, during CRP, they have to continue attacking the DoH traffic to ensure that the DoH can be successfully downgraded to plaintext DNS. On the other hand, the longer CRP is, the more difficult DoH service is to be attacked.

Interval Growth (IG). Between every two consecutive CRP, there will be an interval during which the browser will not send any reconnect request related to DoH. Learning the exact interval, the attacker can pause the attack and sniff the plaintext DNS packets. What's more, by blocking the plaintext DNS queries to resolve the domain of DoH servers, the attacker is able to hold the user always following plaintext DNS. For the attacker, the longer the interval lasts, the stealthier the attack would be (i.e., less downgrade packets to be sent). We also found some browsers use the constant interval but some use interval of growing values, in a linear format. We use IG to differentiate these two cases.

Max Interval (MI). When the interval increases linearly, MI is the maximum value among all intervals. When the interval is constant, MI is the value of the interval.

Analysis of Browser Reactions. We examined the 4 attack methods on every browser through network packet analysis and found *every* combination leads to successful attack, though browsers react differently. According to the official documentation of each browser, browsers other than Firefox follow Chrome in how to configure DoH. Also our empirical analysis shows browsers other than Firefox behave in the identical way. Thus, we separate the browsers into "Firefox" and "Chrome+". The patterns are detailed in Table 2 and Table 3.

As a highlight, we found that *none* of the browsers prompt the user when DoH is downgraded to plaintext DNS. This is problematic as the user might have the perception that his/her DNS communication is still protected and continues to visit the sensitive websites. We also found the extra latency of visiting a website is not prominent when retrying DoH. Therefore, it becomes rather difficult for users to discover the attacks. Regarding CRP, we found the settings are quite diverse, with values ranging from 0.09 seconds to 36.52 seconds, and some even random[2]. In most situations, IG grows linearly and we speculate this is to reduce the unnecessary retries by the browser. While there are four combinations that result in MI of 50 to 65 seconds, we found for other combinations, the browser *will not* attempt to upgrade to DoH within 10 minutes, which is the maximum window we set to test each combination. Leveraging the measurement result of our study, an attacker can make flexible decision about how frequently to attack DoH, which attack method to use, based on his/her resources and victim's environment.

[2]Random in this case means the coefficient of variation (the ratio of standard deviation to the mean of CRP) is greater than 15%.

Attack	BType	CRP	IG	MI
DNS Spoofing	Chrome+	0.09	Linear	N/A
	Firefox	0.10	Constant	65.51
DNS Intercepting	Chrome+	36.52	Linear	N/A
	Firefox	15.01	Linear	50.50
TCP RST Injection	Chrome+	Random	Linear	N/A
	Firefox	Random	Linear	N/A
TCP Intercepting	Chrome+	10.98	Constant	63.84
	Firefox	0.27	Linear	65.25

Table 3: Reaction patterns under downgrade attack. N/A in MI means more than 10 minutes. All numbers are in seconds and are average values from multiple experiments.

4.3 Feedback after Disclosure

We reported our observations via the vulnerability disclosure systems to the 6 browser vendors, together with our suggestions to better manage the threat of downgrade attacks. We have received responses from all of them except Microsoft Edge. Among the ones replied, none consider a fix is necessary in order to prevent DoH from being attacked. According to the reply from Firefox, that DoH is vulnerable under downgrade attacks is an expected feature of Opportunistic Privacy profile. Firefox believes this setting still protects the users against *passive* adversaries most of the time. Chrome claims that the issue is the "intended, designed, and documented behavior" of current Chrome DoH. In fact, Chrome uses plaintext DNS until a DoH resolver can be reliably connected and then upgrade all DNS communications to DoH in opportunistic mode. The communication falls back to plaintext DNS if any connection issue happens. For other browsers whose DoH implementation is provided entirely by the Chromium browser engine, the responses are similar or expected to be.

While given all browsers follow RFC 8310 [35], which has mentioned the possibility of attacks, it is unsurprising that none of the browsers will make a step to address our attacks. However, we are concerned that the user notification is deliberately ignored and *none* of the browser vendors plan to adopt it even though the integration into browser UI should incur moderate effort and overhead (e.g., using Notification API [28,31]). We believe users should be put into the decision loop, so they can choose to avoid sensitive communications (e.g., visiting a controversial website) in a hostile environment when necessary. We have not found any justification for this design choice, but we speculate the reason is to reduce the likelihood that users are annoyed by the notifications. Still, we suggest a discussion among browser vendors and the Internet community should be initiated, because the bar of downgrade attack is relatively low.

5 Countermeasures

Revising DoH implementations. Our evaluation on browsers shows they all configure Opportunistic Privacy profile by default. While this setting avoids service disruption when DoH is unavailable, it is not suitable when a user is in an adversarial network environment. Though strict mode can be turned on for some browser settings, we found they are not easily identifiable to non-technical users, so we suggest the browser UIs be redesigned to make those options more explicit. In the meantime, the browsers that do not support strict mode should incorporate it as soon as possible. As described in Section 4.3, users should better be notified when DoH is disconnected.

To notice, user profiles are also the basic building blocks of DoT [35]. Thus, we recommend future changes should be considered both DoH and DoT.

Revising DoH protocols. We also suggest some components/stages of DoH can be revisited. URI Template [22] is used to define the URI in DoH requests [34]. In most cases, the hostname in URI needs to be resolved via plaintext DNS communications, or Phase 1. This makes DoH dependent on plaintext DNS, which could undermine its security benefit. In particular, we are concerned that a large-scale DoH downgrade attack might be possible, as the censor can leverage its existing middlebox to attack the DNS packet in Phase 1. Unfortunately, a practical fix without dampening the usability of DoH is not easy. (1) By always using IP address of DoH server, one can remove the dependency on DNS, and Firefox has offered this option, but the users have to set up both URI and IP address of a DoH resolver. (2) Another approach is to embed IP directly in URI template as the hostname, but it will prevent the authentication of DoH's identity that looks into SSL certificate of its hostname. Though IP addresses could be authenticated in a SSL certificate [15, 16], it incurs extra configuration from the server owners, which is not yet broadly deployed. One major reason is the lack of clarity on how to authenticate IP's ownership at the webserver's level. Therefore, a new protocol or extension tailored to securing Phase 1 might be needed.

6 Related Work

DNS-over-HTTPS. Though DoH is installed as an RFC only two years ago [34], there have been a number of studies measuring its deployment, understanding its security implications and assessing its impact on the Internet ecosystem. For the measurement works, Lu et al. studied how DoH servers are distributed around the world, their reachability and performance, and the trend in user adoption [27]. Böttger et al. measured the overhead of DoH based on statistical metrics like packet count and load time [8] and Hounsel et al. compared DoT, DoH and plaintext DNS [20]. Deccio et al. measured the prevalence and characteristics of DoH servers and their TCP Fast Open (TFO) support, which is a feature to reduce the latency of DoH [13]. Regarding security, traffic analysis was carried out to understand how likely DoH communications leak users' activities, e.g., whether visiting a website monitored by the adversary [9, 25, 36]. Unfortunately, though protections like padding are applied to DoH packets, the studies show DoH is not resilient against such adversary. Regarding the societal impact, Borgolte et al. measured the effect of DoH on different operators, ISPs, regulations and policies [7]. Our work investigates the security issues of DoH, but starting from a new angle (usage profile) not explored before.

Downgrade attacks. Downgrade attack forces a system to abandon its high-standard security protocol/setting and fall back to an older, weaker one. Downgrade attack has been found possible in TLS [2]. For example, Logjam [1] tricks the server to choose an "export-grade" Diffie-Hellman cipher suite. DROWN [4], on other hand, downgrades a TLS client to use SSLv2 during key exchange. TLS version could also be downgraded to an earlier one [5]. Apart from TLS, downgrade attack is also explored in ARM hardware infrastructure [10], 5G [23] and WPA3 certification [37]. Though we carried out downgrade attacks against DoH, our goal is not to invent a new downgrade algorithm. Instead, we show the existing, relatively simple Denial-of-Service attack like TCP Reset can be used for downgrade attack against DoH, due to the usage profile settings of the browser.

7 Conclusion

In this paper, we studied how browsers implement and configure DoH, especially how they react when DoH is interfered by an active adversary. We tested 4 different network attack vectors on DoH communications of 6 browsers and examined whether DoH can be downgraded to plaintext DNS. Our evaluation shows every combination of browser and attack vector leads to successful attack, and the attack is hard to be spotted, due to the lack of user notifications and relaxed reconnections. Though the issues we found are not explicit vulnerabilities on browsers, given that the browsers all follow the usage profiles RFC, we ask the Internet community to carefully examine the security implications of usage profiles. As temporary fixes, we proposed a few countermeasures that can be quickly applied, e.g., protecting the Phase 1 of DoH communication.

Acknowledgement

We thank our reviewers and our shepherd Drew Springall for their great feedback.

References

[1] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, et al. Imperfect forward secrecy: How Diffie-Hellman fails in practice. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 5–17, 2015.

[2] Eman Salem Alashwali and Kasper Rasmussen. What's in a downgrade? a taxonomy of downgrade attacks in the TLS protocol and application protocols using TLS. In *International Conference on Security and Privacy in Communication Systems*, pages 468–487. Springer, 2018.

[3] Anonymous. The NSA and GCHQ's QUANTUMTHEORY Hacking Tactics. https://theintercept.com/document/2014/03/12/nsa-gchqs-quantumtheory-hacking-tactics/.

[4] Nimrod Aviram, Sebastian Schinzel, Juraj Somorovsky, Nadia Heninger, Maik Dankel, Jens Steube, Luke Valenta, David Adrian, J Alex Halderman, Viktor Dukhovni, et al. DROWN: Breaking TLS using SSLv2. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 689–706, 2016.

[5] Karthikeyan Bhargavan, Christina Brzuska, Cédric Fournet, Matthew Green, Markulf Kohlweiss, and Santiago Zanella-Béguelin. Downgrade resilience in key-exchange protocols. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 506–525. IEEE, 2016.

[6] Philippe Biondi. Scapy: explore the net with new eyes. *Technical report, EADS Corporate Research Center*, 2005.

[7] Kevin Borgolte, Tithi Chattopadhyay, Nick Feamster, Mihir Kshirsagar, Jordan Holland, Austin Hounsel, and Paul Schmitt. How dns over https is reshaping privacy, performance, and policy in the internet ecosystem. *Performance, and Policy in the Internet Ecosystem (July 27, 2019)*, 2019.

[8] Timm Böttger, Felix Cuadrado, Gianni Antichi, Eder Leão Fernandes, Gareth Tyson, Ignacio Castro, and Steve Uhlig. An empirical study of the cost of DNS-over-HTTPS. In *Proceedings of the Internet Measurement Conference*, pages 15–21, 2019.

[9] Jonas Bushart and Christian Rossow. Padding ain't enough: Assessing the privacy guarantees of encrypted dns. *arXiv preprint arXiv:1907.01317*, 2019.

[10] Yue Chen, Yulong Zhang, Zhi Wang, and Tao Wei. Downgrade attack on trustzone. *arXiv preprint arXiv:1707.05082*, 2017.

[11] Richard Clayton, Steven J Murdoch, and Robert NM Watson. Ignoring the great firewall of china. In *International Workshop on Privacy Enhancing Technologies*, pages 20–35. Springer, 2006.

[12] Cloudflare. DNS over HTTPS. https://developers.cloudflare.com/1.1.1.1/dns-over-https, 2020.

[13] Casey Deccio and Jacob Davis. DNS privacy in practice and preparation. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, pages 138–143, 2019.

[14] Frank Denis and Yecheng Fu. Dnscrypt. https://www.dnscrypt.org/, 2015.

[15] Geocerts. Using an ip address in an ssl certificate. https://www.geocerts.com/support/ip-address-in-ssl-certificate.

[16] GlobalSign. Securing a public ip address - ssl certificates. https://support.globalsign.com/ssl/general-ssl/securing-public-ip-address-ssl-certificates.

[17] Google. The chromium projects: Dns over https (aka doh). https://www.chromium.org/developers/dns-over-https.

[18] Christian Grothoff, Matthias Wachs, Monika Ermert, and Jacob Appelbaum. SA's MORECOWBELL: knell for DNS. *Unpublished technical report*, 2017.

[19] Dominik Herrmann, Christian Banse, and Hannes Federrath. Behavior-based tracking: Exploiting characteristic patterns in dns traffic. *Computers & Security*, 39:17–33, 2013.

[20] Austin Hounsel, Kevin Borgolte, Paul Schmitt, Jordan Holland, and Nick Feamster. Comparing the effects of DNS, DoT, and DoH on web performance. In *Proceedings of The Web Conference 2020*, pages 562–572, 2020.

[21] Z Hu, L Zhu, J Heidemann, A Mankin, D Wessels, and P Hoffman. Rfc 7858-specification for dns over transport layer security (tls). http://www.ietf.org/rfc/rfc7858.txt, 2016.

[22] J.Gregorio, R.Fielding, M.Hadley, M.Nottinghamand, and D.Orchard. Uri template. http://www.ietf.org/rfc/rfc6570.txt. RFC 6570, DOI:10.17487/RFC6570, March 2012.

[23] Mohsin Khan, Philip Ginzboorg, Kimmo Järvinen, and Valtteri Niemi. Defeating the downgrade attack on identity privacy in 5g. In *International Conference on Research in Security Standardisation*, pages 95–119. Springer, 2018.

[24] Dae Wook Kim and Junjie Zhang. You are how you query: Deriving behavioral fingerprints from DNS traffic. In *International Conference on Security and Privacy in Communication Systems*, pages 348–366. Springer, 2015.

[25] Carlos López Romera. DNS Over HTTPS traffic analysis and detection. 2020.

[26] Graham Lowe, Patrick Winters, and Michael L Marcus. The great DNS wall of China. *MS, New York University*, 21:1, 2007.

[27] Chaoyi Lu, Baojun Liu, Zhou Li, Shuang Hao, Haixin Duan, Mingming Zhang, Chunying Leng, Ying Liu, Zaifeng Zhang, and Jianping Wu. An End-to-End, Large-Scale Measurement of DNS-over-Encryption: How Far Have We Come? In *Proceedings of the Internet Measurement Conference*, pages 22–35, 2019.

[28] Joseph Medley. Web Push Notifications: Timely, Relevant, and Precise. https://developers.google.com/web/fundamentals/push-notifications, 2020.

[29] Paul Mockapetris. Domain names—implementation and specification. http://www.ietf.org/rfc/rfc1035.txt, 2004.

[30] Mozilla. Firefox dns-over-https. https://support.mozilla.org/en-US/kb/firefox-dns-over-https.

[31] Mozilla. Notification - Web APIs. https://developer.mozilla.org/en-US/docs/Web/API/notification, 2019.

[32] Corey Nachreiner. Anatomy of an ARP poisoning attack. *Retrieved July*, 4:2005, 2003.

[33] Angela Orebaugh, Gilbert Ramirez, and Jay Beale. *Wireshark & Ethereal network protocol analyzer toolkit*. Elsevier, 2006.

[34] P.Hoffman and P.McManus. Dns queries over https (doh). http://www.ietf.org/rfc/rfc8484.txt. RFC 8484, DOI:10.17487/RFC8484, October 2018.

[35] S.Dickinson, D.Gillmor, and T.Reddy. Usage profiles for dns over tls and dns over dtls. http://www.ietf.org/rfc/rfc8310.txt. RFC 8310, DOI:10.17487/RFC8310, March 2018.

[36] Sandra Siby, Marc Juarez, Claudia Diaz, Narseo Vallina-Rodriguez, and Carmela Troncoso. Encrypted DNS –> privacy? a traffic analysis perspective. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*. The Internet Society, 2020.

[37] Mathy Vanhoef and Eyal Ronen. Dragonblood: Analyzing the dragonfly handshake of WPA3 and EAP-pwd. In *Proceedings of the 2020 IEEE Symposium on Security and Privacy-S&P 2020)*. IEEE, 2020.

[38] Wireshark. How to Decrypt 802.11. https://wiki.wireshark.org/HowToDecrypt802.11, 2020.