Title
A Comprehensive Study of DNS-over-HTTPS Downgrade Attack

Author
Huang, Qing

Publication Date
2020

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA,
IRVINE

A Comprehensive Study of DNS-over-HTTPS Downgrade Attack

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Networked Systems

by

Qing Huang

Thesis Committee:
Assistant Professor Zhou Li, Chair
Assistant Professor Alfred Qi Chen
Assistant Professor Ardalan Amiri Sani

2020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank Prof. Zhou Li and Dr. Deliang Chang for their guidance on this thesis. I would like to express my gratitude to Prof. Alfred Qi Chen and Prof Ardalan Amiri Sani for accepting to be my thesis committee. I would also like to thank my parents and my friends for their supporting.

ABSTRACT OF THE THESIS

A Comprehensive Study of DNS-over-HTTPS Downgrade Attack

By

Qing Huang

Master of Science in Networked Systems

University of California, Irvine, 2020

Assistant Professor Zhou Li, Chair

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.

Chapter 1

Introduction

1.1 DNS-over-HTTPS Downgrade Attack

The Domain Name System (DNS) is the phonebook of the Internet. Each device connected to the Internet has a unique IP address and DNS maps human-friendly string-type domain names to machine-friendly numerical IP addresses [18], which other machines use to find the device. 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 [44], 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 [5, 30, 41].

To mitigate these threats and protect DNS's authenticity, confidentiality and integrity, several protocols aiming to transmit DNS packets in encrypted channel are proposed [34, 52, 23]. Among those methods, DNS-over-HTTPS (DoH) [52] is most promising, since it has already been implemented and integrated into most of the popular browsers like Google Chrome [28] and Firefox [45]. It is also offered as a service by large public resolvers like Cloudflare [20]. In

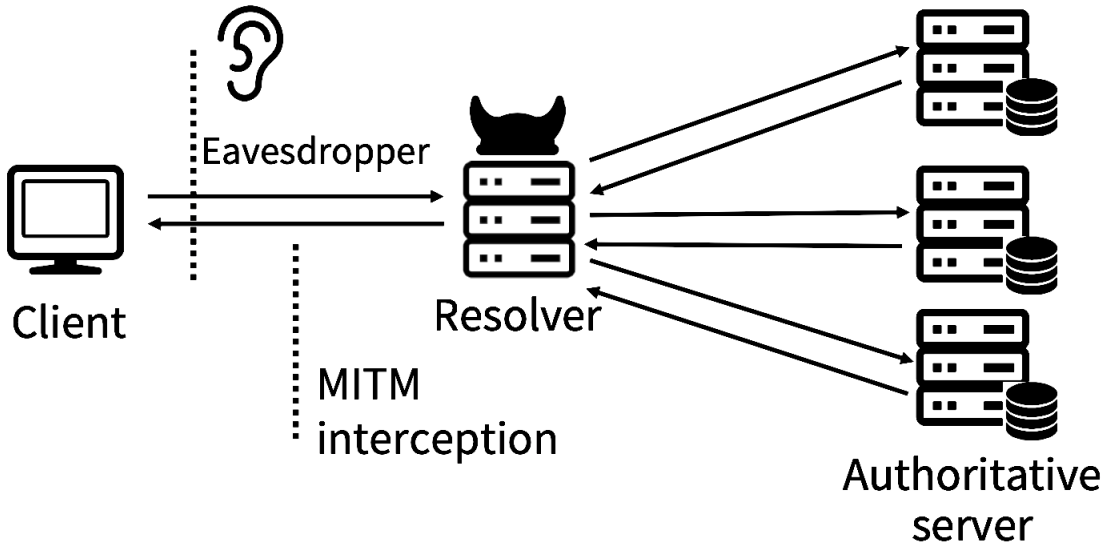

Figure 1.1: DNS Threat. Dotted lines indicate weak points that can be eavesdropped or maliciously attacked

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) [34] which requires a specialized stub resolver, the deployment overhead of DoH is much lower on the client-side.

During 2018, Google and Cloudflare launched a public DoH parser, and Mozilla added DoH support in Firefox, and since September 2019 [29], it will gradually be introduced as the default configuration for Firefox users in the United States [46]. It is precisely because of the extensive support of many DNS resolver vendors and browser vendors that we will use DoH as the entry point for this research. There have been a lot of researches on DoH security analysis, such as the working efficiency of the mechanism, the traffic recognition rate and overhead of DoH [54]. But these are security studies under normal circumstances, and do not consider the performance of DoH or the browser's response mode in the presence of an

attacker, and these studies are more on the DoH mechanism itself rather than considering the browser The strategic level of using DoH in real scenarios. Therefore, in order to fill this gap, we conduct a comprehensive study on the response to downgrade attacks of DoH and browser response measures when facing downgrade attacks.

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 [55] 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 thesis, 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 which are already integrated with 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. And we also discussed the possible countermeasures, protocol improvement and suggestions of best practice to deal with downgrade attack. Our contributions are listed below:

- We perform the *first* study of downgrade attacks on DoH, by systematically enumerat-

ing the attack surface and examining the attack vectors. We find as many downgrade attacks as possible by perform a comprehensive analysis on the process of how mainstream browsers adopted DoH and IETF documentations. We systematically categorize those attacks and note that censoring DoH traffic does not require the censor to deploy new attack schemes.

- We evaluate the attacks in a realistic lab environment and perform all kinds of attacks on several mainstream browsers on several operating systems, find them all vulnerable under the default settings. And users are not notified when DoH is not available and their queries could be compromised. Our finding is downgrade attack is not only feasible but succeeding against all browsers. We also found the reactions of browsers under attack are concerning.

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

1.2 Thesis Outline

The rest of this thesis is organized as follow: Chapter 2 describes previous related research and provides background information. Chapter 3 describes the resolution and communication process of DoH, the adversary model we designed and downgrade attack methods target each phase of resolution and communication process of DoH. Chapter 4 describes the usage profile of DoH defined in RFC 8310 [55] experimental setup and tool flow of the downgrade experiment. This part also provides the experimental result of response pattern of browsers facing downgrade attacks. Feasible countermeasures are proposed in Chapter5. Chapter 6 reviews the thesis work and discusses the advantage and disadvantage of DoH bases on this thesis.

Chapter 2

Background

2.1 Domain Name System (DNS)

The process of DNS resolution involves converting a hostname into a computer-friendly IP address. An IP address is given to each device on the Internet, and that address is necessary to find the appropriate Internet device is used to find a particular home. When a user wants to load a webpage, a translation must occur between what a user types into their web browser and the machine-friendly address necessary to locate the webpage [18]. 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 plain-text, making it vulnerable to be manipulated. The DNS packets between stub resolver and recursive resolver are the major target for attackers, and much information is contained in user's DNS queries, which are transmitted between end-users and RRs. Previous research

have shown that DNS queries from users could be used to track [32, 39] or censor [5, 41] them.

2.2 DNS Privacy Considerations

The privacy issue of DNS has attracted the wide attention of many people in the industry. In document [11], the author mainly put forward some of his concerns and thoughts about DNS privacy issues, and summarized some possible DNS privacy issues which is in spirit of section 8 of document [1]. It is intended to be an analysis of the present situation. The author believes that there is a big difference between DNS data itself and a specific transaction (namely DNS name lookup). DNS data and DNS query results are public within the above range, and there may not be any confidentiality requirements. However, this is not the case for a single transaction or series of transactions. The transaction is not/should not be made public. According to the author's description, the nodes that can leak DNS private information mainly include wire, cache, rouge server and authoritative name servers. But this document only raises the problem; it does not attempt to set requirements (with implicit choices and tradeoffs), nor does it define solutions. We will discuss possible solutions to the problem described here in other background documents. While in [3], Shulman explores the dependencies in DNS and demonstrates the technique of using side channel leakage due to transfer trust, so that information about the target domain can be inferred in encrypted DNS packets. He mentions DNS transmit is public and prevents attacks against DNS by encrypted DNS is very important. He also emphasizes the compatibility with some network Infrastructures and traditional protocols when using DNS encryption [56]. How to handle third party proxies and the support of basic protocols should also be taken into consideration.

2.3 Hypertext Transfer Protocol Security (HTTPS).

Hypertext Transfer Protocol Security (HTTPS) is a secure version of HTTP, which is the main protocol used to send data between web browsers and websites. HTTPS is encrypted to improve the security of data transmission. This is especially important when users transmit sensitive data (for example, by logging into a bank account, email service, or health insurance provider) [19].Any website, especially those that require login credentials, should use HTTPS. In modern web browsers (such as Chrome), sites that do not use HTTPS are marked differently from sites that use HTTPS. Look for the green padlock in the URL bar to indicate that the page is safe.

2.4 Enhancing DNS Privacy

In the first version of DNS, security was not a major consideration, and for many years DNS traffic was sent over the (untrusted) network. In the past few years, concerns about security and privacy have prompted the emergence of solutions that make DNS traffic resistant to eavesdropping and tampering. Several studies have empirically demonstrated how to abuse the openness of DNS traffic for censorship [6] [51] and surveillance [31] [16]. Early efforts included protocols such as DNSSEC and DNSCrypt. DNSSEC prevents the use of digital signatures to manipulate DNS data. However, it does not provide confidentiality. DNSCrypt is an open source work that provides confidentiality and authenticity. However, due to lack of standardization, it has not been widely adopted. In 2016, the IETF approved DNS-over-TLS (DoT)as a standard tracking protocol. The client establishes a TLS session with the recursive resolver (usually on the TCP: 853 port standardized by IANA), and exchanges DNS queries and responses through an encrypted connection. To amortize costs, the TLS

session between the client and the recursive DNS resolver usually remains active and can be reused for multiple queries. Queries pass through this channel in the same way as in unencrypted DNS. In the standardized DoH in 2018, the local DNS resolver establishes an HTTPS connection with the recursive resolver and encodes the DNS query into the body of the HTTP request. The Ministry of Health considers using HTTP/2 server push mechanism. This enables the server to preemptively push DNS responses that may follow DNS lookups, thereby reducing communication delays. Compared with DoT, DoT uses a dedicated TCP port for DNS traffic, so it is easy to monitor and block, while DoH lookups can be sent along non-DNS traffic using existing HTTPS connections (but may be blocked At the IP level).

2.5 DNS-over-HTTPS (DoH)

DNS services have penetrated into every corner of the Internet. Initially, DNS data packets were designed to be transmitted on the Internet in an unencrypted form. However, this design is not secure. Attackers can intercept and analyze DNS data packets to damage the security and privacy of Internet users. To mitigate the privacy issues of DNS, large vendors such as Google and Cloudflare have adopted solutions to encrypt DNS lookups, such as DNS over Https (DoH). 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. Though DoH is installed as an RFC only two years ago [52], 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 [42]. Böttger et al. measured the overhead of DoH based on statistical metrics like packet count and load time [13] and Hounsel et al. compared DoT, DoH and plaintext DNS [33]. 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 [21]. 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 [57, 14, 40]. 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 [10].this thesis investigates the security issues of DoH, but starting from a new angle (usage profile) not explored before.

2.6 DNS over TLS (DoT)

DNS over TLS is defined in RFC 7858 [34]. The DoT protocol works for queries and responses between stub clients and recursive servers. The list of DoT implementations maintained by the DNS Privacy Project includes stubs and recursive resolvers, forwarders, command line tools, and browsers [54]. Initiating DoT is done by establishing a connection on a well-known port, the client and server can expect and agree to negotiate a TLS session to protect the channel [34]. While in article [53], the author conduct an investigation on the information leakage of DNS-over-TLS (DoT). DoT traffic can be identified by the method they proposed with a false negative rate of less than 17% and a false positive rate of less than 0.5% when DNS messages are not padded and information leakage can still occur even

when DoT messages are padded. Their study shows that, to a certain extent, DoT cannot completely protect the privacy of users' DNS information. Attackers can still obtain information about user activity by analyzing DoT traffic. The DoT specification points out the possibility of such attacks against encrypted DNS

2.7 DNS Query Name Minimization

The idea of DNS query name minimization is to minimise the amount of data sent from the DNS resolver to the authoritative name server. In the example in the previous section, sending "What are the NS records for .com?" would have been sufficient (since it will be the answer from the root anyway). The rest of this section describes the recommended way to do QNAME minimisation – the way that maximises privacy benefits (other alternatives are discussed in the appendices). Instead of sending the full QNAME and the original QTYPE upstream, a resolver that implements QNAME minimisation and does not already have the answer in its cache sends a request to the name server authoritative for the closest known ancestor of the original QNAME. QNAME minimisation is compatible with the current DNS system and therefore can easily be deployed; since it is a unilateral change to the resolver, it does not change the protocol. (Because it is a unilateral change, resolver implementers may do QNAME minimisation in slightly different ways.) The benefits of minimizing QNAME are obviously to reduce the risk of using authoritative name servers. However, minimizing the amount of data sent can also partially solve the privacy violations of wired sniffers and servers. (Of course, encryption is a better way to defend against network sniffer, but unlike QNAME minimization, it changes the protocol and cannot be deployed unilaterally. Moreover, the impact of QNAME minimization on network sniffer depends on the sniffer Is it on the DNS path?) QNAME minimization provides zero protection to the recursive resolver, and the recursive resolver still sees that the complete request comes from the stub resolver

[12]. But DoH has replaced the recursive resolver to a certain extent, which guarantees safety to a greater extent.

2.8 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 [4]. For example, Logjam [2] tricks the server to choose an "export-grade" Diffie-Hellman cipher suite. DROWN [7], on other hand, downgrades a TLS client to use SSLv2 during key exchange. TLS version could also be downgraded to an earlier one[8]. Apart from TLS, downgrade attack is also explored in ARM hardware infrastructure[15], 5G[38] and WPA3 certification[58]. 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.

Chapter 3

DoH Downgrade Attacks

3.1 DNS Basic Principle

Before we begin to explain the principle of DoH and the design of the downgrade attack method, we need to have a clear understanding of the DNS parsing process and principle.

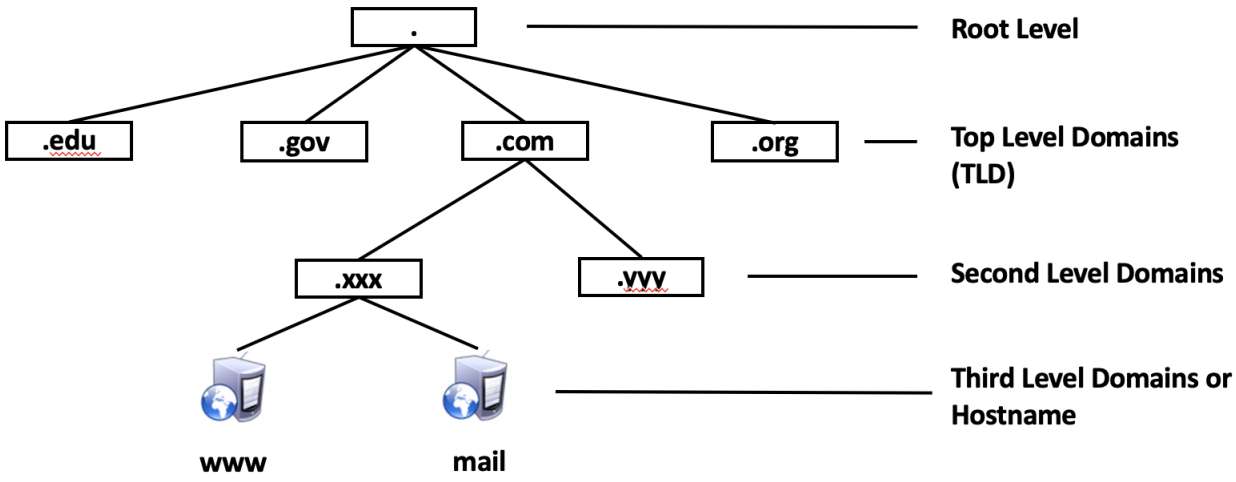

Figure 3.1: DNS Hierarchy. The general hierarchical structure of DNS is tree structure, from top to bottom, it should be the root level domain, top level domain, second level domain and third level domain or hostname

Figure 3.1 shows the hierarchy level of a standard domain name system. DNS uses a hierarchy to manage its distributed database system. The DNS hierarchy, also called the domain name space, is an inverted tree structure. The DNS tree has a single domain at the top of the structure called the root domain. A period or dot (.) is the designation for the root domain. Below the root domain are the top-level domains that divide the DNS hierarchy into segments [49]. The host is located at the lowest level of the domain name space, it is a specific computer, such as www, mail, are specific computer names

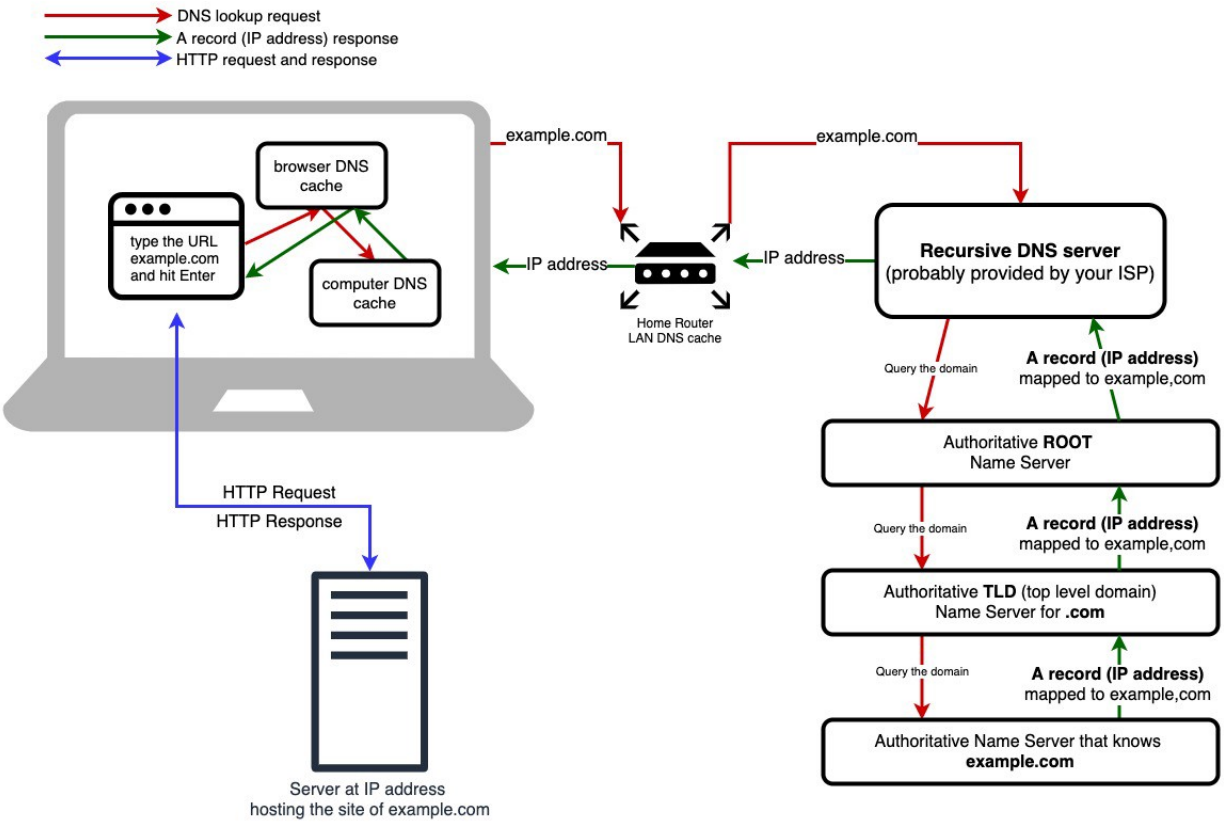

Figure 3.2: DNS Resolution Process. The red pointers represent DNS lookup request traffic, the green pointers indicate the DNS response traffic while blue pointers represent the HTTP request and response traffic.

As shown in 3.2, there are 4 DNS servers involved in loading a webpage [18]: **DNS recursive resolver** - A DNS recursive server is a server designed to receive queries from client computers through applications such as web browsers. Then, usually, the recursive program

is responsible for making other requests to satisfy the client's DNS query. **Root nameserver** - The root server is the first step in converting (resolving) a human-readable hostname into an IP address. Usually, it is used as a reference to other more specific locations. **TLS nameserver** - The top-level domain server (TLD) is the next step in searching for a specific IP address, and it hosts the last part of the hostname (in example.com, the TLD server is "com"). **Authoritative nameserver** - The authoritative name server is the last stop in the name server query. If the authoritative name server can access the requested record, it will return the IP address of the requested host name to the DNS recursor (librarian) that made the initial request.

There are typically 8 steps in a DNS lookup. 1) A user types domain name into a web browser and the query travels into the Internet and is received by a DNS recursive resolver. 2) The resolver then queries a DNS root nameserver (.). [18] 3) The root server then responds to the resolver with the address of a Top Level Domain (TLD) DNS server (such as .com or .net), which stores the information for its domains. 4) When searching for this domain name, the request is pointed toward the TLD. 5) The resolver then makes a request to the TLD. 6) The TLD server then responds with the IP address of the domain's nameserver. 7) Lastly, the recursive resolver sends a query to the domain's nameserver. 8) The IP address is then returned to the resolver from the nameserver. The DNS resolver then responds to the web browser with the IP address of the domain requested initially.

As TLS of HTTPS provides strong privacy and security protection, for the existing DNS attacks to succeed, and running DoH relies on the security of the underlying HTTP transport. This mitigates classic amplification attacks for UDP-based DNS. Therefore, 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.

3.2 Process of DoH Communication

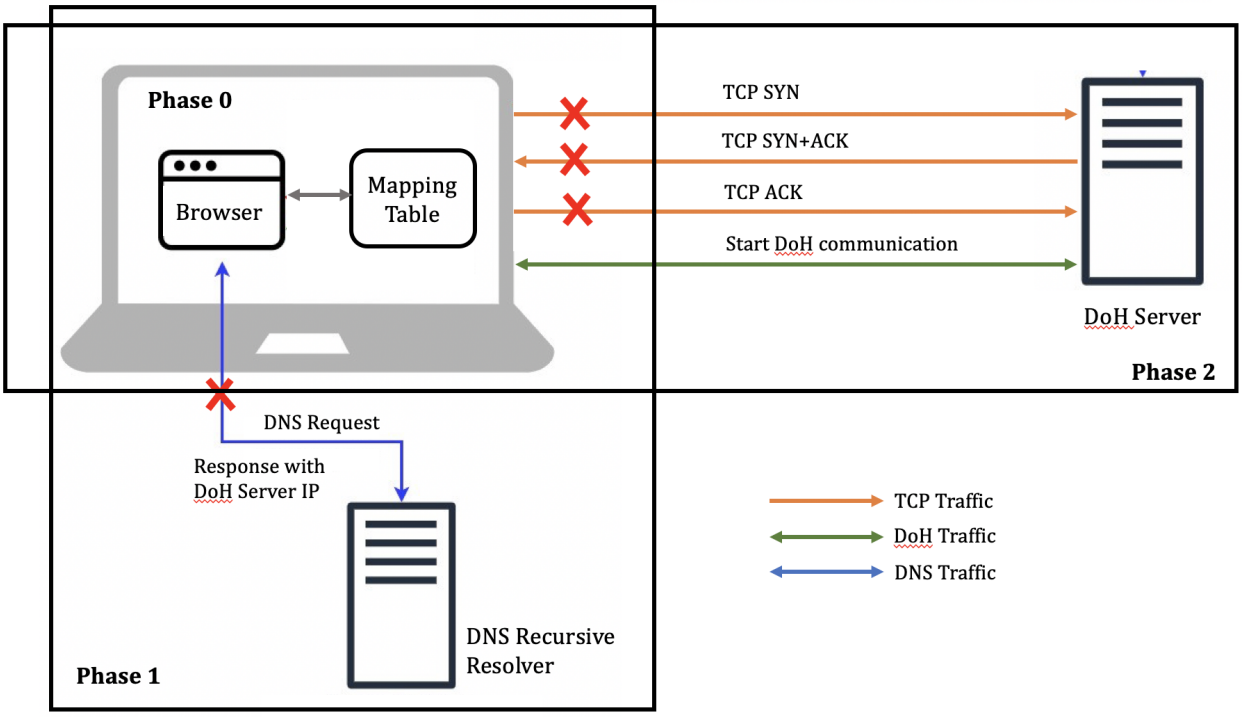

Figure 3.3: DoH resolution process. Crosses represent the attack surface.

We simulated the operation of the browser DoH in real usage scenarios, and used Wireshark to capture specific DoH traffic for DoH traffic analysis. Based on our analysis on popular browsers, DoH communication usually follows two phases as illustrated in Figure 3.3.

Phase 1: URI Resolution. URI used in HTTP requests of DoH is defined by a URI template[52]. The DoH client is configured with a URI template, which describes how to

construct a URL for parsing. The configuration, discovery, and update of the URI template are completed out-of-band from the protocol. Please note that the configuration may be manual (for example, the user enters the URI template of "options" in the user interface) or automatic (for example, the URI template provided in the response of DHCP or similar protocols). 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). To notice, although the official claims DoH is a security mechanism that improves unencrypted DNS to protect user privacy, it still uses unencrypted DNS to resolve the domain name of DoH server in the first phase of DoH resolution process. 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. It provides us with an vulnerability that can be exploited.

Phase 2: Connection & Communication. The browser side establishes a secure HTTPS 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. DoH servers implement both the POST and GET methods. An attacker able to compromise this phase could force the browser to fall back to plaintext DNS. When using the POST method, the DNS query is included as the message body of the HTTP request, and the Content-Type request header field indicates the media type of the message. POSTed requests are generally smaller than their GET equivalents. Using the GET method is friendlier to many HTTP cache implementations. In order to maximize the friendliness of HTTP caching, DoH clients use a media format that includes the ID field from the DNS message header (for example, "application/dns-message"), and DNS ID of 0 should be used in each DNS request. HTTP associates the request with the response, thereby eliminating the need for an ID of the media type (for example, "application/dns-message"). Using changed DNS IDs may cause seman-

tically equivalent DNS queries to be cached separately. [52]

Additionally, A DoH client uses configuration to select the URI, and thus the DoH server, that is to be used for resolution. defines how HTTPS verifies the DoH server's identity. [25] An example URI specifying HTTP/TLS is: *https://www.example.com/ smith/home.html* [25]. 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.

3.3 Adversary Model

Having identify the process of DoH resolution, this section we describes several kinds of attack methods we performed target on each phase of the resolution process, aiming to downgrade the DoH to plain-text DNS. we begin by discussing the potential adversary model. In this part, we assume that the attacker can be located anywhere on the communication network. The attacker is a malicious entity. The target of the attack is the entity (ie, the device or network connected to the network) that is intended to suffer the consequences of the attack. In this context The middle finger makes the higher version service fall back to the lower version DNS. Many participants on the network can perform downgrade attack on a certain user. Here firstly we try to comprehensively define the adversary models and give some real-life examples.

The attacker's goal is to force the encrypted DoH falling back to plaintext. Before the attack,

there is a very important constraint that the attacker must have the ability to sniff the DoH traffic from client side. In this paper, we assume two types of adversaries based on their capabilities of manipulating network packets.

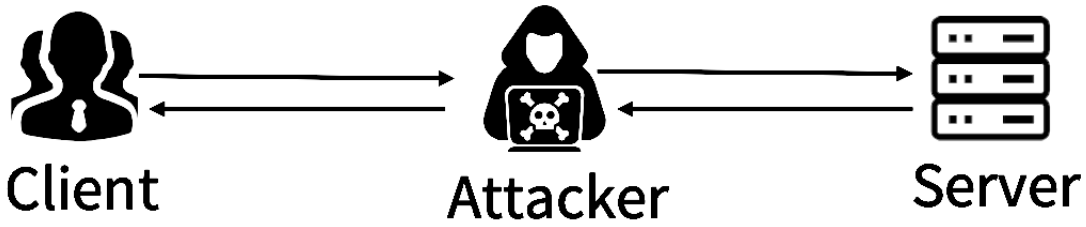

Figure 3.4: In-path Attcker

3.3.1 In-path Attackers.

In-path attackers (also known as man in the middle) can intercept and modify data packets between legitimate communicating entities. The attacker in the path is directly located on the normal communication path (by accessing the nodes on the path, or directly placing himself on the path). As figure 3.4 shows, 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 [48] to redirect the victim traffic to the attacker's machine and act as a rogue middlebox.

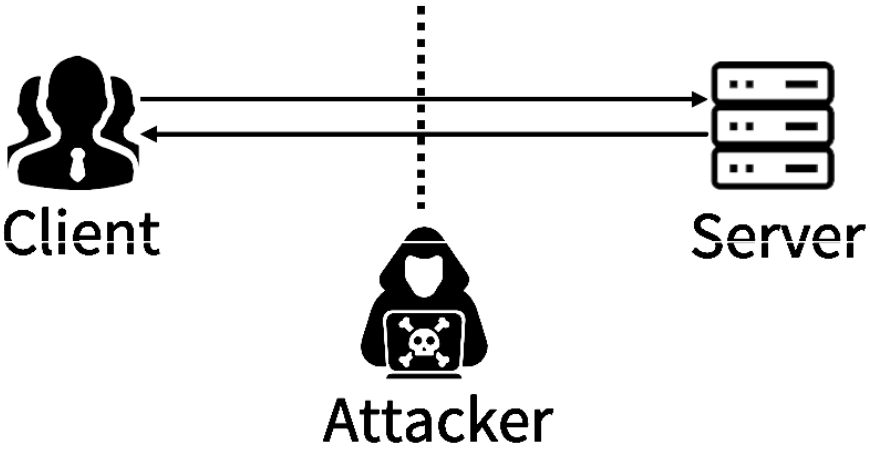

Figure 3.5: On-path Attcker

3.3.2 On-path Attackers.

On-path attacker is located outside the location path, but tries to deviate (or obtain data packets) between the communicating entities [35]. 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[41], 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 [59]. 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. Noticeable, An in-path attacker could be a network gateway. since DoH may affect the services provided by ISP to a certain extent, such as parental protection, which is very inconvenient for ISP, so they may tend to prohibit users from using DoH. While an on-path attacker could be another user sharing LAN with the victim. There are many ways to allow attackers under the same LAN to sniff the traffic of the victim host, regardless of whether the wireless network is encrypted or not.

Alternatively, the on-path attacker under the same LAN with the victim can also use ARP cache poisoning or other method to redirect the traffic to the attacker machine. In this case, the attacker actually play the rule of in-path attacker. From which we can see, on-path attacker and in-path attacker can be converted to each other. The ARP poisoning attack works as follow: The attacker must have access to the network. They scan the network to determine the IP addresses of at least two devices. Suppose these devices are workstations and routers. Attackers use spoofing tools such as ARP spoofing or Driftnet to send fake ARP responses. The fake response indicates that the correct MAC address of the two IP addresses belonging to the router and the workstation is the MAC address of the attacker. This makes routers and workstations unable to connect to the attacker's computer and cannot connect to each other. The two devices will update their ARP cache entries, and from this point onwards, communicate with the attacker instead of directly communicating with each other. The attacker is now conducting all communications in secret.

3.4 Attack Method

Based on the workflow of DoH described in Section 3.2, 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. Before we discuss the downgrade attack methods that we designed, we need to know two specific protocols first, domain name system and transmission control protocol:

Domain name system All communications inside of the domain protocol are carried in

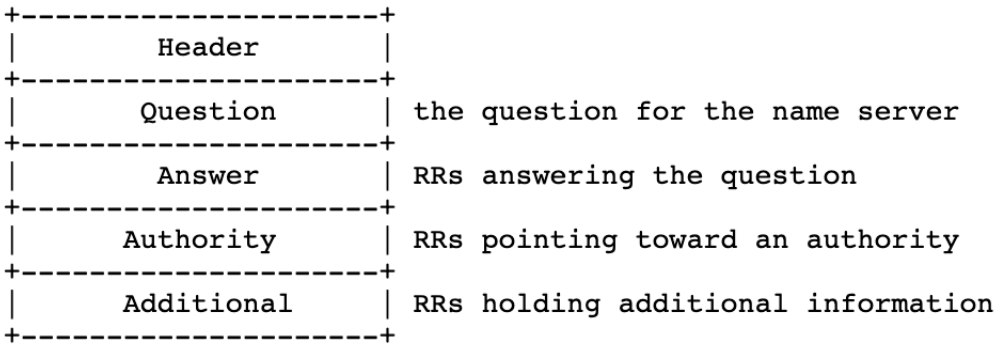

Figure 3.6: DNS Packet Format. [44]

a single format called a message. The top level format of message is divided into 5 sections (some of which are empty in certain cases) shown in Figure 3.6. The header includes fields that specify which of the remaining sections are present, and also specify whether the message is a query or a response, a standard query or some other opcode, etc.

The names of the sections after the header are derived from their use in standard queries. The question section contains fields that describe a question to a name server. These fields are a query type (QTYPE), a query class (QCLASS), and a query domain name (QNAME). The last three sections have the same format: a possibly empty list of concatenated resource records (RRs). The answer section contains RRs that answer the question; the authority section contains RRs that point toward an authoritative name server; the additional records

[1]Only for Chrome browser.

section contains RRs which relate to the query, but are not strictly answers for the question [44]. In this paper, we mainly use the fields after the beginning fields. The most important ones are the question and answer fields. The details of using and tampering with the relevant data fields in the message will be detailed later.

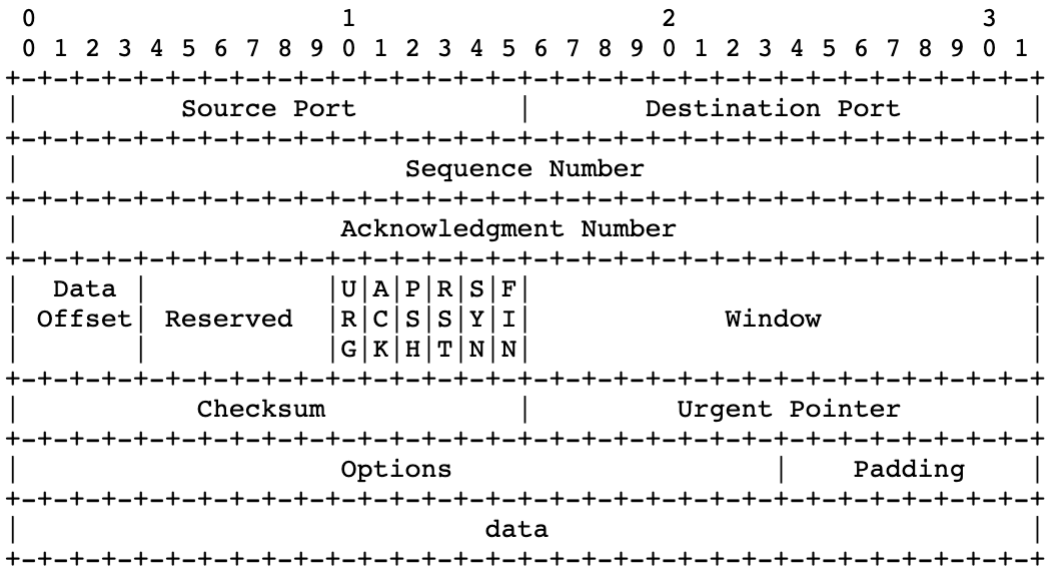

Figure 3.7: Transmission Control Protocol Header Format. [22]

Transmission control protocol For the TCP protocol we are most concerned about its header field. TCP segments are sent as internet datagrams. The Internet Protocol header carries several information fields, including the source and destination host addresses. A TCP header follows the internet header, supplying information specific to the TCP protocol. This division allows for the existence of host level protocols other than TCP [22]. Next, I will introduce in detail the fields we will use during the downgrade attack. Sequence Number is 32 bits long. The sequence number of the first data octet in this segment (except when SYN is present). If SYN is present the sequence number is the initial sequence number (ISN) and the first data octet is ISN+1.Acknowledgment Number is 32 bits long and if the ACK control bit is set this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established this is always sent. Control

Bits is 6 bits long. What we need to focus on is the RST flag in the control field. SYN is synchronize sequence numbers. ACK is acknowledgment field significant. A device will send a RST in response to receiving a packet for a closed socket. TCP RST packet is the remote side telling you that the connection on which the previous TCP packet is sent is not recognized, maybe the connection has closed, maybe the port is not open, and something like these.

Below, we will provide more detail about four attack methods we have implemented and examine these attack can be done by what kind of adversary. Table 3.1 shows different adversary models and corresponding downgrade attack methods they can exploit based on their abilities.

Attack Method	On-path Attacker	In-path Attacker
DNS traffic intercepting		√
DNS cache poisoning	√	√
TCP traffic intercepting		√
TCP reset injection	√	√

Table 3.1: Attack method and corresponding adversary model.

3.4.1 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 4.2.

3.4.2 DNS Cache Poisoning

On-path attackers target Phase 1. Domain Name System (DNS) poisoning and spoofing are types of cyberattack that exploit DNS server vulnerabilities to divert traffic away from legitimate servers towards fake ones. Once you've traveled to a fraudulent page, you may be puzzled on how to resolve it — despite being the only one who can. You'll need to know exactly how it works to protect yourself. DNS spoofing and by extension, DNS cache poisoning are among the more deceptive cyberthreats. Without understanding how the internet connects you to websites, you may be deceived into thinking a website itself is hacked. In some cases, it may just be your device. Even worse, cybersecurity suites can only stop some of the DNS spoof-related threats [37]. DNS cache poisoning in this case 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. This method only requires the attacker to have the ability to listen to the network traffic.

3.4.3 TCP Traffic Interception

In-path attacker targets Phase 2. The most obvious way to tamper DoH service is to intercept TCP traffic in DoH's 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.

3.4.4 TCP Reset Injection

More subtly, tampering the TCP traffic from connection & communication phase can be achieved by an on-path attacker. In the packet flow of a TCP connection, each packet contains a TCP header. Each of these headers contains a bit called a "reset" (RST) flag. In most data packets, this bit is set to 0 and has no effect; however, if this bit is set to 1, it indicates to the receiving computer that the computer should stop using the TCP connection immediately; it should not use the connection identification number (called Port) then send any packets, and discard any other received packets with headers indicating that they belong to the connection. TCP reset basically terminates the TCP connection immediately. This can be a useful tool when used as designed. A common application is a situation where the computer (computer A) crashes during a TCP connection. The computer on the other end (computer B) will continue to send TCP packets because it does not know that computer A has crashed. After restarting computer A, it will receive packets from the old pre-crash connection. Computer A does not have the context of these packets, nor can it know how to process these packets, so it may send a TCP reset to Computer B. This reset lets Computer B know that the connection no longer works. The user on computer B can now try other connections or take other measures. 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 then send a forged packet containing the TCP reset to one or two endpoints. The header in the forged packet must incorrectly indicate that it came from the endpoint, not the forger. This information includes the endpoint IP address and port number. Each field in the IP and TCP headers must be set to a convincing forged value for a pseudo reset to trick the endpoint into closing the TCP connection. A properly formatted fake TCP reset can be a very effective way to interrupt any TCP connection that the forger can monitor. 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 [41, 17]. Assume a censor adds domain names listed in Table 4.2 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.

Chapter 4

Attack Evaluation

In this section, we evaluate our four downgrade attack methods designed based on the DoH resolution and communication process regarding different vectors, which including different DoH servers, different browsers that support DoH services and different operating systems. In this evaluation, we not only considered the effectiveness of our attacking methods but also measured the behavior of various browsers that support DoH after being attacked, through network traffic analysis, and found out their specific response pattern facing each attacking method.

4.1 Usage Profile of DoH

According to the requirements of authentication and encryption, Usage profile provides different levels of mitigation methods for DNS client attacks, regardless of context. The usage profile is a different concept from the usage strategy or usage model; the usage strategy or usage model may stipulate which configuration file should be used in a specific context,

Usage Profile	Connection	Passive Attacker	Active Attacker
Strict	A, E	P	P
Opportunistic	A, E	P	P
Opportunistic	E	P	N, D
Opportunistic		N, D	N, D

Table 4.1: Attack Protection by Usage Profile and Type of Attacker. P == Protection; N == No protection; D == Detection is possible; A == Authenticated connection; E == Encrypted connection. [55].

using a specific set of DNS servers or referring to other external factors. According to [55], there are two specific usage profiles (Strict Privacy and Opportunistic Privacy) for DoH that provide improved levels of mitigation for the attacks described above compared to using only cleartext DNS. We presented a generalized discussion of usage profiles by separating the usage profile, which is based purely on the security properties it offers the user, from the specific mechanism or mechanisms that are used for DNS server authentication. The profiles described are below:

- **Strict Privacy profile** Strict privacy profile needs to encrypt the connection and successfully authenticate the DNS server; this not only reduces passive eavesdropping, but also reduces client redirection (if an authenticated encrypted connection is not provided, DNS service is not provided).
- **Opportunistic Privacy profile** Opportunistic privacy profile will try but does not require encryption and successful authentication; therefore, it cannot provide limited mitigation measures for this type of attack, or even no mitigation measures, but it can maximize the chances of DNS service.

In order to compare these two usage profiles, Table 4.1 shows three possible outcomes of a successful strict privacy profile and an opportunistic privacy profile. In the best case of opportunistic privacy profiles (authenticated and encrypted connections), it is equivalent to strict privacy profiles. In the worst case, it is equivalent to plain text. Customers who use

opportunistic privacy profiles should try the best case, but in order to get a response, they may fall back to the intermediate case and eventually the worst case. One of the reasons for not trying to use all available privacy-enabled DNS servers and falling back is whether delay is more important than mitigating the attack. Therefore, opportunistic privacy profiles provide different protections based on the type of connection actually used (including no mitigation at all). The "detection" described below is only possible when such connection information is available. This can be especially useful if new connections to a particular server were not encrypted when all previous connections were encrypted. Similarly, if the user is notified that an encrypted but unauthenticated connection is used, the user can detect that the connection may be actively attacked. In other words, for the case where no protection is provided against the attacker (N), it can be detected that an attack may be occurring (D).

With Strict Privacy, the DNS client MUST NOT commence sending DNS queries until at least one of the privacy-enabling DNS servers becomes available. In this case, even though the connection failed, DoH client won't use plaintext DNS still. But we found that all browsers implement DoH following opportunistic privacy profile, thus, we will discuss opportunistic privacy profile in detail as follow. Our downgrade attack depends on the fallback logic of the client, the available authentication information, and the capabilities of the DNS server. Opportunity privacy is defined as using plaintext as a basic communication security strategy, and negotiating encryption and authentication, and applying it to communications when available [24]. The default baseline is clear text rather than full protection. When all peers do not support full protection, the OS protocol will not shrink from full protection; on the contrary, the OS protocol aims to use the maximum protection available. (At a certain point in time for a particular application or protocol, almost negligible peers can support encryption. At that time, the baseline security may be improved from plaintext to always requiring encryption, and only need time Authentication). Since in opportunistic privacy

profile, the DoH client has a mechanism to fall back to a lower version when a connection error occurs, so we can use this feature to achieve our downgrade attack method.

4.2 Experiment Setup

DoH Server	URI	Domain name
Google	https://dns.google/dns-query	dns.google
Cloudflare	https://cloudflare-dns.com/dns-query	chrome.cloudflare-dns.com(*) cloudflare-dns.com
Quad9	https://dns.quad9.net/dns-query	dns.quad9.net
Umbrella/OpenDNS	https://doh.opendns.com/dns-query	doh.opendns.com
CleanBrowsing	https://doh.cleanbrowsing.org/doh/family-filter	doh.cleanbrowsing.org
Comcast	https://doh.xfinity.com/dns-query	doh.xfinity.com
DNS.SB	https://doh.dns.sb/dns-query	doh.dns.sb

Table 4.2: Domain names of DoH resolvers. (*) means only for Chrome

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 4.3, which are all popular and supporting DoH service. 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 [50] to eavesdrop victim's traffic and Scapy-2.4.3 [9] 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 set up a DoH server configuration on their DoH client side, in this case, is browsers that integrated DoH service first and then visit several random websites. Both Phase 1 (URI Resolution) and Phase 2 (DoH Connection & Communication) are tested under same circumstance. Though we examined different DoH servers listed in Table 4.2, 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	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 4.3: Browser DoH settings. "Config" indicates how to set DoH resolver's URI. "Profile" is the type of usage profile [55]. "BType" categorizes how a browser reacts when facing DoH Downgrade Attack. "Notif" is whether browser notifies its user when DoH is not usable.

Browser Settings. Table 4.3 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 [28, 45]. 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[55], 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.

4.3 Browser Reaction under Attack

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 4.1. 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.

4.3.1 Correlation Parameter

In this experiment, we mainly used the method of analyzing the network data packets captured by Wireshark to simulate the behavior pattern of the browser in response to DoH downgrade attack. We will discuss the methods of calculating relevant parameters for the four degraded attack methods designed above.

The response pattern of the browser is mainly related to the measures taken by the browser vendors to deal with the situation that the network traffic cannot be connected normally at the code implementation level. The DoH downgrade attack is essentially a process of denying DoH service traffic. We have no way to know the specific code implementation of the browser, so we consider them as blackbox and attempt to understand their response patterns through our experimental analysis. Figure 4.1 shows the browsers' behaviour after the DoH connection fails. We explain the detail and some terminologies defined in this article as follow.

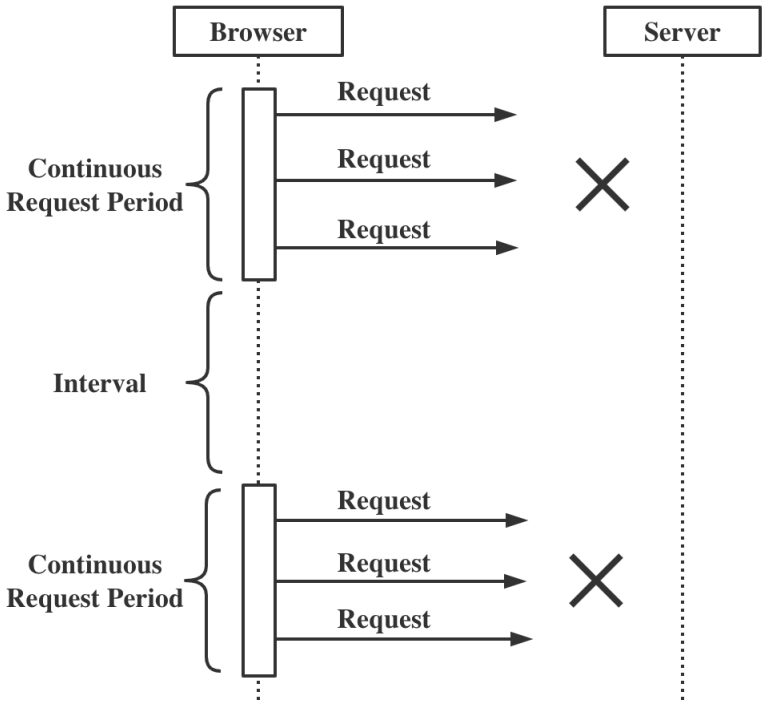

Figure 4.1: Reconnection Process after Failure.

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.

Furthermore, as the reconnect request still fails, the growth of the interval time will follow a certain pattern, which may be to keep a constant value unchanged or increase to a certain value with a linear growing trend.

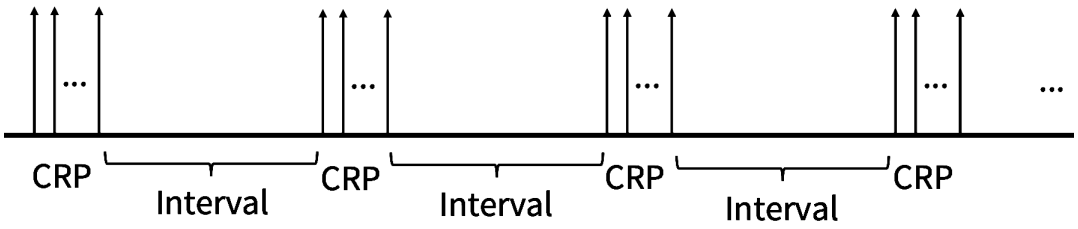

Figure 4.2: Constant Interval Growth.

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.

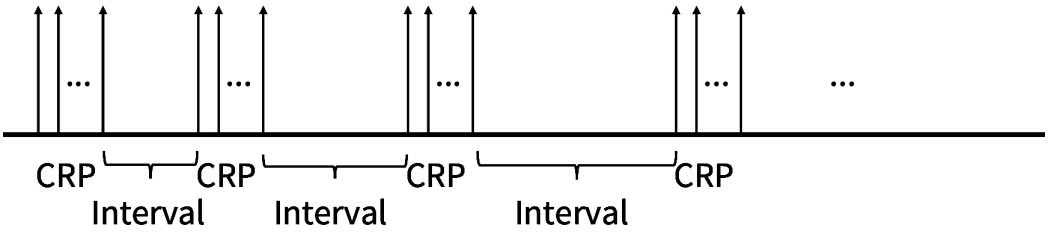

Figure 4.3: Linear Interval Growth.

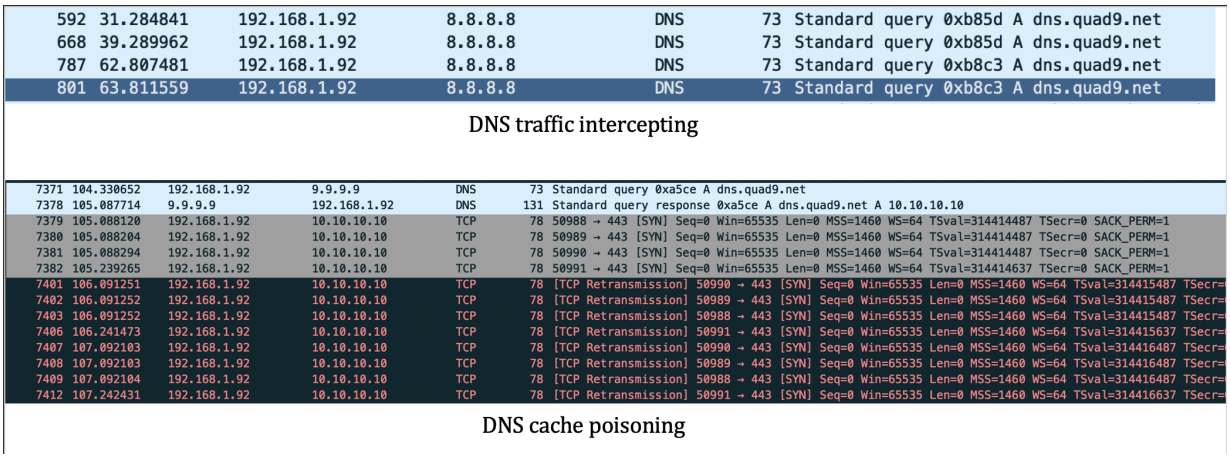

Figure 4.4: Packet capture of DNS-related downgrade attacks.

4.3.2 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 4.3 and Table 4.4. From this point it is not difficult to understand why Firefox has a slightly different response pattern compared to others.

Packet capture of DNS-related downgrade attacks. Figure 4.4 shows the network packets captured by Wireshark when we perform a DNS-related downgrade attack on the browser. In this case, every time the browser sends out a DNS request, it is regarded as a re-connection behavior after Doh fails. We define the interval of dns requests sent continuously in a short period of time as continuous request period.

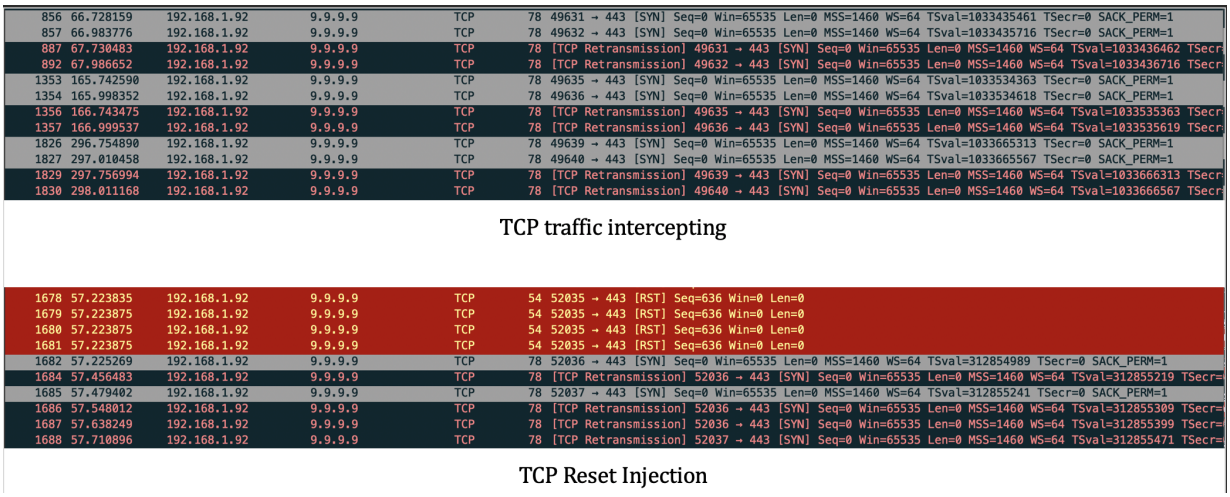

Figure 4.5: Packet capture of TCP-related downgrade attacks.

Packet capture of TCP-related downgrade attacks. The browser's response to attacks on TCP-related traffic is shown in Figure 4.5. The picture above is a screenshot of Wireshark that directly blocks specific TCP traffic, while second one is the result of TCP reset injection attack. For traffic analysis of TCP-related attacks, every TCP request packet with SYN field sent by the browser can be defined as a re-connection attempt. In this context, continuous request period is the time period for continuously sending SYN packets.

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.

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 4.4: Reaction patterns under downgrade attack. When the CV (coefficient of variation) of CRP is greater than 15%, we label it as Random. N/A in MI means more than 10 minutes. All numbers are in seconds and are average values from multiple experiments.

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[1]. 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.

Users think they are using a secure DoH, and they have no awareness of the existence of the attacker. An attacker can easily steal user's privacy without being discovered by the victim. To a certain extent, if there is no security prompt, the difficulty of attacking DoH is similar to the difficulty of attacking DNS. In other words, DoH service does not provide much protection against DNS attack.

As mentioned above, the goal of our downgrade attack on the DoH service is not just to reject the use of the DoH service, but to force the DoH to fall back to the plain-text DNS.

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

In this case, if an attacker attack the DoH service in strict privacy profile, it will make the user unable to access the webpage normally. After the user fails to access, it is equivalent to prompting the user in disguise. Thus, for our DoH downgrade attacks, the attack effectiveness of the opportunistic privacy profile is much higher than the strict privacy profile.

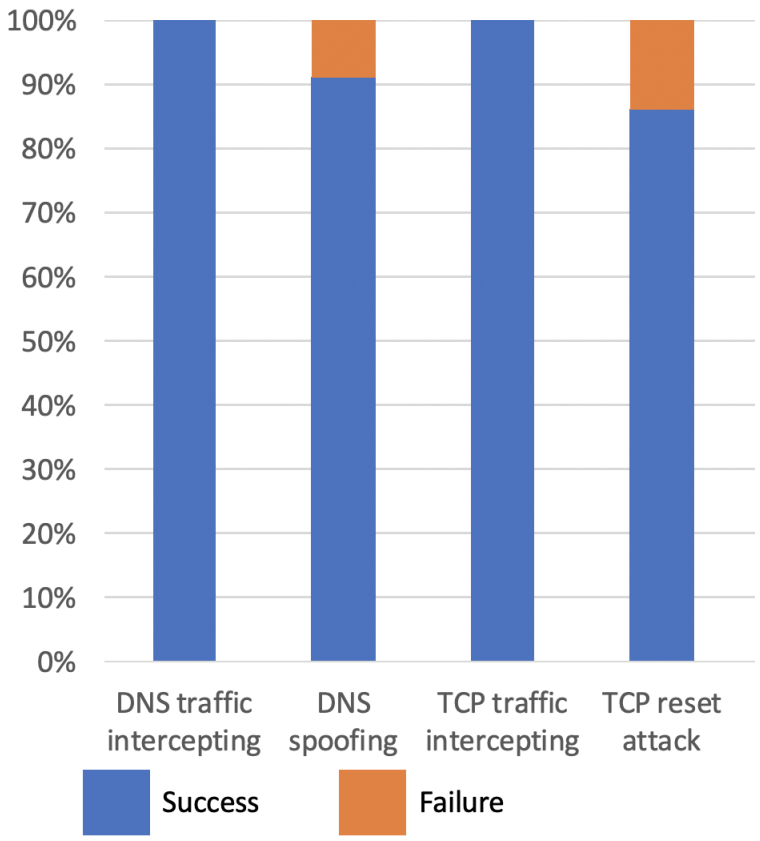

Figure 4.6: Downgrade attack success rate.

Downgrade attack success rate. In order to effectively detect the success rate of the downgrade attack method in actual operation, we will qualitatively analyze the situation of traffic data packet capture without using DoH and the situation of traffic data packet capture using DoH but without attack methods as a reference. Through this analysis method, we display the success rate results in figure 4.6. We calculated the percentage of the number of plaintext DNS packages captured under the attack and the number of plaintext DNS packages when DoH was not used to get the success rate of downgrade attack. it can be

obviously seen that directly blocking DoH-related traffic has a hundred percent downgrade rollback rate. The two attack methods of DNS cache poisoning and TCP reset injection may cause a small probability of failure because the speed of sending forged packets may be lower than the response speed of real packets (continue to use encrypted DoH service)

4.4 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. Both upgrade and downgrade are designed to be silent, without notifying users. 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 [55], 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 [43, 47]). 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.

We speculate that the reason why browser vendors ignore this issue is that they only try to protect "most requests" even though there are still exceptions that can be completed using classic DNS. However, based on our experience above, it is actually very simple to implement DoH downgrade attack. The attacker only needs to add one step to the traditional DNS attack so that they can easily obtain user's privacy information.

Chapter 5

Countermeasures

5.1 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 4.4, users should better be notified when DoH is disconnected. To notice, user profiles are also the basic building blocks of DoT [55]. Thus, we recommend future changes should be considered both DoH and DoT. It's helpful to draft a similar standard for DoH since many browsers already implemented this in their browser. Our evaluation shows DoH downgrade attack is practical and should be mitigated. In today's setting, it is possible for a user to have "encrypted DNS" turned on, but still be using plaintext DNS. The user cannot even verify if it's using DoH to access a website. It's counterintuitive. So

from implementation's perspective, 1) a browser could allow user to choose "strict mode" or "DoH only mode" if desirable. For now, only Firefox has configuration about this. Even for Firefox itself, the setting about "DoH only mode" is under "about:configs" thus it is very unconvenient to change. 2) in "opportunistic mode", user should be notified when fallback happens. For now, no browser notifies its users about this, leaving them exposed to potential surveillance and censorship without their knowledge.

5.2 Revising DoH Protocols

We also suggest some components/stages of DoH can be revisited. URI Template [36] is used to define the URI in DoH requests [52]. 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. This dependency also makes DoH vulnerable when facing downgrade attack even by plaintext DNS manipulation. 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. In addition to being inconvenient, it is also difficult for users to directly confirm the credibility of the link between URI template and IP address. (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 [26, 27], 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.

Chapter 6

Conclusion

Pros and Cons of DoH DoH is still in an experimental state, and there are many old DNS infrastructures that do not support encryption, so what are the advantages and disadvantages for users? DoH protects the privacy of users to a large extent, and solves some of the following problems:

- DNS resolution requests and resolution results cannot be decrypted when they are routed through the intermediate. End-to-end encryption between the client and the resolution server is achieved, so that privacy is not leaked in the middle.
- The two-way request and result data can not be tampered with because of the existence of the encryption algorithm, which prevents phishing attacks and denial of service attacks using DNS.

However, there will still be some unsolved problems of DoH service:

- Trusted recursive resolution servers are publicly available. ISPs can block these IP addresses and allow users to transfer to other mirror servers. After all, the current client browser upgrade process is slow. Rolling-back to plaintext DNS is possible.

- The firewall rules cannot be triggered by determining the user's access to the domain name, which can cause large amount of malicious domains evade detection.
- Since all browsers supporting DoH service use opportunistic privacy profile by default, this makes downgrade attacks possible, and in our analysis, the browser attempts to reconnect is in a low frequency, which provides a certain convenience to the attacker.

In this thesis, 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. These results will influence the evolution of DNS privacy standards and can provide inspiration for DNS vendors' security when using the next version.

Bibliography

[1] B. A. J. P. J. M. M. H. R. S. A. Cooper, H. Tschofenig. Privacy considerations for internet protocols. `http://www.ietf.org/rfc/rfc6973.txt`, 2015.

[2] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. 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.

[3] H. S. . Affiliations. Pretty bad privacy: Pitfalls of dns encryption. 2014.

[4] E. S. Alashwali and K. 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.

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

[6] Anonymous. Towards a comprehensive picture of the great firewall's DNS censorship. *USENIX Security Symposium. USENIX Association)*, 2014.

[7] N. Aviram, S. Schinzel, J. Somorovsky, N. Heninger, M. Dankel, J. Steube, L. Valenta, D. Adrian, J. A. Halderman, V. Dukhovni, et al. DROWN: Breaking TLS using SSLv2. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 689–706, 2016.

[8] K. Bhargavan, C. Brzuska, C. Fournet, M. Green, M. Kohlweiss, and S. Zanella-Béguelin. Downgrade resilience in key-exchange protocols. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 506–525. IEEE, 2016.

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

[10] K. Borgolte, T. Chattopadhyay, N. Feamster, M. Kshirsagar, J. Holland, A. Hounsel, and P. 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.

[11] S. Bortzmeyer. Dns privacy considerations. http://www.ietf.org/rfc/rfc7626.txt, 2015.

[12] S. Bortzmeyer. Dns query name minimisation to improve privacy. http://www.ietf.org/rfc/rfc7816.txt, 2016.

[13] T. Böttger, F. Cuadrado, G. Antichi, E. L. Fernandes, G. Tyson, I. Castro, and S. Uhlig. An empirical study of the cost of DNS-over-HTTPS. In *Proceedings of the Internet Measurement Conference*, pages 15–21, 2019.

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

[15] Y. Chen, Y. Zhang, Z. Wang, and T. Wei. Downgrade attack on trustzone. *arXiv preprint arXiv:1707.05082*, 2017.

[16] M. E. Christian Grothoff, Matthias Wachs. NSA's MORECOWBELL: Knell for DNS. *Unpublished technical report*, 2017.

[17] R. Clayton, S. J. Murdoch, and R. N. Watson. Ignoring the great firewall of china. In *International Workshop on Privacy Enhancing Technologies*, pages 20–35. Springer, 2006.

[18] Cloudflare. What is dns? how dns works. https://www.cloudflare.com/learning/dns/what-is-dns/, 2019.

[19] Cloudflare. What is https? https://www.cloudflare.com/learning/ssl/what-is-https/, 2019.

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

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

[22] M. del Rey. Transmission control protocol. http://www.ietf.org/rfc/rfc793.txt, 1981.

[23] F. Denis and Y. Fu. Dnscrypt. https://www.dnscrypt.org/, 2015.

[24] V. Dukhovni. Opportunistic security: Some protection most of the time. http://www.ietf.org/rfc/rfc7435.txt, 2013.

[25] E.Rescorla. Http over tls. https://tools.ietf.org/html/rfc2818. RFC 2818, DOI:10.17487/RFC2818, May 2000.

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

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

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

[29] Google. Google public dns over https (doh) supports rfc 8484 standard. https://security.googleblog.com/2019/06/google-public-dns-over-https-doh.html, 2019.

[30] C. Grothoff, M. Wachs, M. Ermert, and J. Appelbaum. SA's MORECOWBELL: knell for DNS. *Unpublished technical report*, 2017.

[31] S. Guha and P. Francisn. Identity trail: Covert surveillance using DNS. *Privacy Enhancing Technologies Symposium (PETS)*, 2007.

[32] D. Herrmann, C. Banse, and H. Federrath. Behavior-based tracking: Exploiting characteristic patterns in dns traffic. *Computers & Security*, 39:17–33, 2013.

[33] A. Hounsel, K. Borgolte, P. Schmitt, J. Holland, and N. Feamster. Comparing the effects of DNS, DoT, and DoH on web performance. In *Proceedings of The Web Conference 2020*, pages 562–572, 2020.

[34] 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.

[35] L. Iannone. Locator/id separation protocol (lisp) threat analysis. http://www.ietf.org/rfc/rfc7835.txt, 2016.

[36] 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.

[37] Kaspersky. What is dns cache poisoning and dns spoofing? https://usa.kaspersky.com/resource-center/definitions/dns, 2019.

[38] M. Khan, P. Ginzboorg, K. Järvinen, and V. Niemi. Defeating the downgrade attack on identity privacy in 5g. In *International Conference on Research in Security Standardisation*, pages 95–119. Springer, 2018.

[39] D. W. Kim and J. 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.

[40] C. López Romera. DNS Over HTTPS traffic analysis and detection. 2020.

[41] G. Lowe, P. Winters, and M. L. Marcus. The great DNS wall of China. *MS, New York University*, 21:1, 2007.

[42] C. Lu, B. Liu, Z. Li, S. Hao, H. Duan, M. Zhang, C. Leng, Y. Liu, Z. Zhang, and J. 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.

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

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

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

[46] Mozilla. dns over https (doh) – testing on beta. `https://blog.mozilla.org/futurereleases/2018/09/13/dns-over-https-doh-testing-on-beta`, 2019.

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

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

[49] Novell.com. Dns hierarchy. `https://www.novell.com/documentation/dns_dhcp/?page=/documentation/dns_dhcp/dhcp_enu/data/behdbhhj.html`, 2019.

[50] A. Orebaugh, G. Ramirez, and J. Beale. *Wireshark & Ethereal network protocol analyzer toolkit*. Elsevier, 2006.

[51] F. L. Paul Pearce, Ben Jones and R. Ensafi. Global measurement of dns manipulation. . *USENIX Security Symposium. USENIX page 22*, 2017.

[52] 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.

[53] C. C. H. W. Rebekah Houser, Zhou Li. An investigation on information leakage of dns over tls. 2019.

[54] C. D. Sandra S, Marc J. Encrypted dns =¿privacy? a traffic analysis perspective. 2019.

[55] 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.

[56] H. Shulman. Pretty Bad Privacy Pitfalls of DNS Encryption. https://www.ietf.org/proceedings/93/slides/slides-93-irtfopen-1.pdf, 2014.

[57] S. Siby, M. Juarez, C. Diaz, N. Vallina-Rodriguez, and C. 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.

[58] M. Vanhoef and E. 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.

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