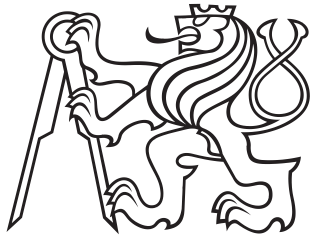


Master's Thesis



Czech  
Technical  
University  
in Prague

**F3**

Faculty of Electrical Engineering  
Department of Telecommunication Engineering

## Credibility of Encrypted DNS Traffic

**Bc. Jan Šimůnek**

Supervisor: Ing. Bc. Marek Neruda, Ph.D.

Supervisor–specialist: doc. Ing. Lukáš Vojtěch, Ph.D.

Field of study: Electronics and Communications

Subfield: Communication Networks and Internet

May 2021



## I. Personal and study details

Student's name: **Šimůnek Jan** Personal ID number: **457155**  
Faculty / Institute: **Faculty of Electrical Engineering**  
Department / Institute: **Department of Telecommunications Engineering**  
Study program: **Electronics and Communications**  
Specialisation: **Communications Networks and Internet**

## II. Master's thesis details

Master's thesis title in English:

**Credibility of Encrypted DNS Traffic**

Master's thesis title in Czech:

**Důvěryhodnost šifrovaného DNS provozu**

Guidelines:

Describe the encrypted DNS protocol with a focus on the standardized variants DNS over HTTPS (DoH-RFC 8484) and DNS over TLS (DoT-RFC 7858). Design a procedure for detecting these DoH and DoT. Design and implement security analysis of these protocols implemented in publicly available DNS resolvers. Focus on adhering to appropriate cryptographic algorithms and procedures defined by the RFC.

Bibliography / sources:

- [1] S. Siby, M. Juarez, C. Diaz, N. Vallina-Rodriguez, and C. Troncoso, "Encrypted DNS --> Privacy? A Traffic Analysis Perspective," Proceedings 2020 Network and Distributed System Security Symposium, 2020, doi: 10.14722/ndss.2020.24301.  
[2] "DNS Queries over HTTPS (DoH)," IETF.org, 2018. <https://tools.ietf.org/html/rfc8484> (accessed Feb. 03, 2021).  
[3] "Specification for DNS over Transport Layer Security (TLS)," IETF.org, 2016. <https://tools.ietf.org/html/rfc7858> (accessed Feb. 03, 2021).

Name and workplace of master's thesis supervisor:

**Ing. Marek Neruda, Ph.D., Department of Telecommunications Engineering, FEE**

Name and workplace of second master's thesis supervisor or consultant:

**doc. Ing. Lukáš Vojtěch, Ph.D., Department of Telecommunications Engineering, FEE**

Date of master's thesis assignment: **05.02.2021** Deadline for master's thesis submission: **21.05.2021**

Assignment valid until: **30.09.2022**

Ing. Marek Neruda, Ph.D.  
Supervisor's signature

Head of department's signature

prof. Mgr. Petr Páta, Ph.D.  
Dean's signature

## III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

\_\_\_\_\_  
Date of assignment receipt

\_\_\_\_\_  
Student's signature



## Acknowledgements

I would like to thank my supervisor Ing. Bc. Marek Neruda, Ph.D. and supervisor-specialist doc. Ing. Lukáš Vojtěch, Ph.D. for their kind guidance throughout my studies. They supported me, gave me an opportunity to fulfill my interests and always provided me with valuable advice.

I must express my gratitude to my parents, sister and grandparents, who unconditionally supported me and motivated me throughout my whole life.

I would also like to acknowledge my girlfriend for her valuable ideas and patience.

## Declaration

I hereby declare that the presented work was carried out independently and that I have listed all information sources used in accordance with the Methodical Guidelines on Maintaining Ethical Principles During the Preparation of Higher Education Theses. Furthermore, I declare that the borrowing and publishing of my thesis or part of it is allowed with the agreement of department.

In Prague, 20. May 2021

.....

## Abstract

The master's thesis describes standardized encrypted DNS (DNS over TLS and DNS over HTTPS) and analyses their security implementation in open recursive resolvers. The detailed specification of DNS with its extension mechanism is introduced and TLS protocol encapsulating exchanged messages is examined. Multiple aspects of encrypted DNS such as increased security and potential misuse by cybercriminals or service providers are described. The performed survey showed that implementations lack of padding which makes encrypted traffic analysis more difficult. It was also discovered that reconnaissance attack and software version identification through CHAOS class was possible. In most of the cases was proved that TLS and its negotiated cryptography parameters sufficiently followed security recommendations. However in many cases the certificates showed poor quality to establish trust between a server and a client.

**Keywords:** DNS, DNS over HTTPS, DNS over TLS, TLS, cryptography, privacy

**Supervisor:** Ing. Bc. Marek Neruda, Ph.D.

## Abstrakt

Tématem diplomové práce je popis standardizovaných variant šifrovaného DNS, tedy DNS over TLS a DNS over HTTPS a jejich následná bezpečnostní analýza ve veřejně dostupných rekurzivních resolvech. Práce se zabývá detailním popisem protokolu DNS a protokolu TLS zapouzdřující výměnu zpráv. Zmíněny jsou jak aspekty zvýšené bezpečnosti, tak potenciální možnosti zneužití šifrovaného DNS, ať už z pohledu útočníka nebo z pohledu poskytovatele služby. Provedený průzkum poukazuje na nedostatky implementace tzv. paddingu zvyšující odolnost vůči analýzám šifrovaného provozu a možnosti identifikace verze softwaru systému skrze CHAOS třídu. Z pohledu vrstvy TLS je ve většině případů správně dodržen postup dohody kryptografických algoritmů. Výjimku tvoří certifikáty, které se pro koncové klienty mohou velmi často jevit jako nedůvěryhodné.

**Klíčová slova:** DNS, DNS over HTTPS, DNS over TLS, TLS, kryptografie, soukromí

**Překlad názvu:** Důvěryhodnost šifrovaného DNS provozu

# Contents

0.1 List of Abbreviations . . . . .	2	5.3 Differences between TLS 1.2 and TLS 1.3 . . . . .	28
<b>1 Introduction</b>	<b>5</b>	<b>6 Encrypted DNS</b>	<b>31</b>
<b>Part I</b>		6.1 DNS over TLS . . . . .	32
<b>Theoretical Part</b>		6.2 DNS over HTTPS . . . . .	33
<b>2 The Domain Name System</b>	<b>9</b>	6.3 Privacy Consideration . . . . .	35
2.1 Architecture . . . . .	9	6.3.1 Client-Query Mapping . . . . .	35
2.1.1 Root Name Server . . . . .	10	6.4 Security Issues . . . . .	36
2.1.2 TLD Name Server . . . . .	10	6.4.1 Correlation Analysis . . . . .	36
2.1.3 Authoritative Name Servers . . . . .	11	6.4.2 Encrypted DNS in Enterprise Environment . . . . .	37
2.2 Resolution . . . . .	11	6.5 Performance of Encrypted DNS . . . . .	37
2.2.1 Recursive Resolver . . . . .	11		
2.2.2 Stub Resolver . . . . .	11	<b>Part II</b>	
2.2.3 Name Lookup . . . . .	11	<b>Practical Part</b>	
<b>3 DNS Protocol</b>	<b>13</b>	<b>7 Methodology of Encrypted DNS</b>	
3.1 Resource Record Type . . . . .	13	<b>Data Collection and Analysis</b>	<b>43</b>
3.2 Message Structure . . . . .	14	7.1 Data Collection . . . . .	44
3.3 Extension Mechanisms for DNS (EDNS0) . . . . .	16	7.2 Analysis of DNS Layer . . . . .	45
3.3.1 EDNS0 Padding . . . . .	17	7.2.1 Initial Extraction and EDNS0 Parameters . . . . .	45
3.3.2 EDNS0 Client Subnet . . . . .	17	7.2.2 Server Fingerprinting . . . . .	48
<b>4 Security of DNS Architecture</b>	<b>19</b>	7.3 Analysis of TLS Layer . . . . .	49
4.1 DNS Reflection and Amplification Attacks . . . . .	19	<b>8 Automated Diagnostic for DNS over Encryption</b>	<b>53</b>
4.2 DNS Flood Attack . . . . .	20	8.1 Overview . . . . .	53
4.3 DNS Spoofing . . . . .	20	8.2 Procedure of Analysis . . . . .	55
4.4 DNS Surveillance . . . . .	21	8.2.1 Data Output . . . . .	58
<b>5 Transport Layer Security Protocol</b>	<b>23</b>	<b>9 Results of Measurement</b>	<b>61</b>
5.1 TLS Protocol Overview . . . . .	23	9.1 Collected Data . . . . .	61
5.2 Handshake Protocol . . . . .	24	9.2 DNS Layer . . . . .	63
5.2.1 Key Exchange and Authentication . . . . .	25	9.3 TLS Layer . . . . .	70
5.2.2 X.509 Certificate . . . . .	26	<b>10 Conclusion</b>	<b>75</b>
		<b>Bibliography</b>	<b>77</b>

## Figures

2.1 DNS Tree Structure [2] . . . . .	10	8.1 Architecture of Diagnostic for DNS over Encryption (D2E) . . . . .	54
2.2 Name query for web page access [5] . . . . .	12	8.2 User interface with arguments description . . . . .	55
3.1 DNS message structure, RFC 1035 [8] . . . . .	14	8.3 Flowchart for resp mode . . . . .	56
3.2 Resource Record format, RFC 1035 [8] . . . . .	16	8.4 Flowchart for chaos mode . . . . .	57
3.3 EDNS(0) format, RFC 6891 [9] .	16	8.5 An output of analysis in resp mode . . . . .	58
4.1 Spoofing Attack [18] . . . . .	21	8.6 An example of 1.1.1.1 JSON record . . . . .	60
5.1 Client-Server handshake [27] . . .	25	9.1 The most extended providers of DoT grouped by CN . . . . .	62
5.2 Message sequence for TLS 1.2 with (EC)DHE key-exchange [29] . . . . .	26	9.2 Length of DoT messages . . . . .	64
5.3 Certificate verification with Openssl . . . . .	28	9.3 Length of DoH messages . . . . .	64
6.1 Encrypted DNS query message .	32	9.4 Latency of different DoT message size . . . . .	66
6.2 DNS over TLS communication .	33	9.5 Latency of different DoH message size . . . . .	66
6.3 DoH with HTTP/2 GET message [41] . . . . .	34	9.6 Software implementation of discovered DoT servers . . . . .	69
6.4 DoH with HTTP/2 POST message [41] . . . . .	34	9.7 Software implementation of discovered DoH servers . . . . .	69
6.5 TCP Fast Open [56] . . . . .	38	9.8 Digital signature latency . . . . .	73
6.6 Performance of encrypted DNS [47] . . . . .	39		
7.1 Internet-wide scan for port 853 .	44		
7.2 DNS query over TLS with required recursion and DNSSEC . . . . .	47		
7.3 DNS over TLS query with ECS .	48		
7.4 CHAOS Class and server fingerprinting . . . . .	49		
7.5 List of cipher suites offered by KDIG . . . . .	50		
7.6 List of elliptic curves and signature algorithms . . . . .	51		



## Tables

3.1 RRs table overview .....	14
5.1 TLS 1.3 cipher suites .....	29
8.1 Minimum Requirements .....	54
8.2 Fields and description of JSON object .....	59
9.1 Discovered IPv4 addresses .....	62
9.2 Discovered URLs .....	63
9.3 Messages containing padding ...	65
9.4 DNSSEC support in encrypted DNS .....	67
9.5 ECS support in encrypted DNS.	67
9.6 TLS version across encrypted DNS .....	70
9.7 Negotiated algorithms with DoT servers .....	71
9.8 Negotiated algorithms with DoH servers .....	71
9.9 Key exchange algorithms .....	72
9.10 Digital Signature Algorithms ..	73
9.11 Encryption and hash algorithms	74






---

## 0.1 List of Abbreviations

<b>AD</b> .....	Authenticated Data
<b>AEAD</b> .....	Authenticated Encryption with Associated Data
<b>AES</b> .....	Advanced Encryption Standard
<b>AES-CBC</b> ....	Advanced Encryption Standard-Cipher Block Chaining
<b>API</b> .....	Application Interface
<b>ARPANET</b> ..	Advanced Research Projects Agency NETwork
<b>AS</b> .....	Autonomous System
<b>ASN</b> .....	Autonomous System Number
<b>BGP</b> .....	Border Gateway Protocol
<b>BIND</b> .....	Berkeley Internet Name Domain
<b>CA</b> .....	Certificate authority
<b>CBC</b> .....	Cipher Block Chaining
<b>ccTLD</b> .....	country code Top Level Domain
<b>CD</b> .....	Checking Disabled
<b>CDN</b> .....	Content Delivery Network
<b>CLI</b> .....	Command Line Interface
<b>CN</b> .....	Common Name
<b>CPU</b> .....	Central Processing Unit
<b>CRL</b> .....	Certificate Revocation List
<b>cURL</b> .....	Client Uniform Resource Locator
<b>D2E</b> .....	Diagnostic for DNS over Encryption
<b>DDOS</b> .....	Distributed Denial of Service
<b>DES</b> .....	Data encryption Standard
<b>DH</b> .....	Diffie Hellman
<b>DHCP</b> .....	Dynamic Host Configuration Protocol
<b>DHE</b> .....	Diffie Hellman Ephemeral
<b>DN</b> .....	Distinguished Name
<b>DNS</b> .....	Domain Name System
<b>DNSKEY</b> ....	Domain Name System Public Key
<b>DNSSEC</b> ....	Domain Name System Security Extensions
<b>DO</b> .....	Domain Name System Security Extensions OK
<b>DOH</b> .....	DNS over HTTPS
<b>DOS</b> .....	Denial of Service
<b>DOT</b> .....	DNS over TLS

<b>DPI</b> .....	Deep Packet Inspection
<b>DS</b> .....	Delegation Signer
<b>EC</b> .....	Elliptic Curve
<b>ECDHE</b> .....	Elliptic Curve Diffie Hellman Ephemeral
<b>ECDSA</b> .....	Elliptic Curve Digital Signature Algorithm
<b>ECS</b> .....	Extension Mechanisms for Domain Name System Client Subnet
<b>EDNS</b> .....	Extension Mechanisms for Domain Name System
<b>EU</b> .....	European Union
<b>FQDN</b> .....	Fully Qualified Domain Name
<b>FTP</b> .....	File Transfer Protocol
<b>GCM</b> .....	Galois Counter Mode
<b>GDPR</b> .....	General Data Protection Regulation
<b>gTLD</b> .....	generic Top Level Domain
<b>HKDF</b> .....	HMAC Key Derivation Function
<b>HTTP</b> .....	Hypertext Transfer Protocol
<b>ICANN</b> .....	Internet Corporation for Assigned Names and Numbers
<b>ID</b> .....	Identification
<b>IETF</b> .....	Internet Engineering Task Force
<b>IETF</b> .....	Internet Engineering Task Force
<b>IMAP</b> .....	Internet Message Access Protocol
<b>IP</b> .....	Internet Protocol
<b>ISP</b> .....	Internet Service Provider
<b>JSON</b> .....	JavaScript Object Notation
<b>MD5</b> .....	Message-Digest algorithm
<b>ML</b> .....	Machine Learning
<b>NIST</b> .....	National Institute of Standards and Technology
<b>NSA</b> .....	National Security Agency
<b>NSEC</b> .....	Next Secure
<b>NUKIB</b> .....	National Cyber and Information Security Agency
<b>OCSP</b> .....	OnlineCertificate Status Protocol
<b>OS</b> .....	Operating System
<b>PKI</b> .....	Public Key Infrastructure
<b>PRF</b> .....	Pseudo Random Function
<b>QUIC</b> .....	Quick User Datagram Protocol Internet Connections
<b>RA</b> .....	Recursive Available
<b>RD</b> .....	Recursive Desired



<b>RFC</b> .....	Request for Comments
<b>RR</b> .....	Resource Record
<b>RRSIG</b> .....	Resource Record Signature
<b>RSA</b> .....	Rivest, Shamir, Adleman
<b>RSA-PSS</b> ....	Rivest, Shamir, Adleman Probabilistic Signature Scheme
<b>SAN</b> .....	Subject Alternative Name
<b>SMTP</b> .....	Simple Mail Transfer Protocol
<b>SSL</b> .....	Secure Sockets Layer
<b>sTLD</b> .....	sponsored Top Level Domain
<b>TCP</b> .....	Transport Control Protocol
<b>TFO</b> .....	Transmission Control Protocol Fast Open
<b>TLD</b> .....	Top Level Domain
<b>TLS</b> .....	Transport Layer Security
<b>TR</b> .....	Truncation
<b>TTL</b> .....	Time To Live
<b>UDP</b> .....	User Datagram Protocol
<b>UK</b> .....	United Kingdom
<b>UNIX</b> .....	Uniplexed Information and Computing System
<b>URI</b> .....	Uniform Resource Locator
<b>URL</b> .....	Uniform Resource Locator
<b>USA</b> .....	United States of America
<b>XML</b> .....	Extensible Markup Language



# Chapter 1

## Introduction

The Domain Name System (DNS) is an essential concept for the Internet and local network connectivity. DNS maintains records that include the most significant address space translations. In 1983, DNS was proposed, the Internet was in early-stage at that time and security was an insignificant factor. DNS queries were transferred in plaintext form and no protection against hijacking and surveillance was possible. Almost 40 years later, security and privacy concerns are highly discussed topics and many protocols are redesigned to ensure confidentiality and integrity. DNS protocol is one of them, it lacks security mechanisms and some countermeasures are needed. Accordingly, proposed designs were introduced with additional encryption and two of them were standardized in Request For Comments (RFC) publication as DNS over TLS (DoT) and DNS over HTTPS (DoH). Security issues seem to be overcome but encrypted DNS protocols bring some more challenges to deal with such as more client-specific information and exploitation by cybercriminals. Traditional DNS protocol is still a major protocol to query resolvers, but adoption of encrypted form is expected to be increased.

In the theoretical part of the thesis, DNS and its extension mechanisms will be introduced with emphasis on security issues that encountered traditional implementation based on RFC 882. TLS (Transport Layer Security) will be presented, because of its importance in encryption layer of DoT and DoH. Proposed and standardized protocols will be examined with a focus on security and privacy. The examination will include potential traffic analysis, protocol misuse and privacy issues introduced with the new protocol structure.

In the practical part of the thesis, the procedure of identification of publicly available DoT and DoH resolvers will be described and performed. Further analysis of DNS and its encrypted layer will be reviewed with an automated diagnostic tool. In order to operate with an extensive amount of data, the tool will provide an appropriate output for further data processing. The evaluation of data will be realized with the focus on

privacy, security recommendations and RFC standards.





**Part I**

**Theoretical Part**



## Chapter 2

# The Domain Name System

DNS is a hierarchical distributed database that maintains records for Internet and private network services. It lists names and their corresponding identifiers represented by IP (Internet Protocol) address. Computer communication is based on IP protocol and peers are identified by IPv4 addresses e.g. 212.111.222.222 or even more complex IPv6 addresses e.g. 2001:4860:4860::8888. The notation is not convenient for a human to remember, therefore translation between IP address space and hostname was needed. An example of DNS in practice is a lookup for `www.example.com` in a web browser which translates the hostname also referred more accurately as FQDN (Fully Qualified Domain Name) to the corresponding IP address without a user notice.

DNS has its predecessor in ARPANET (Advanced Research Projects Agency NETWORK) days and `host.txt` file which function the same as DNS does. It shares mapping between IP address and hostname. The remnant of host file is still present in `/etc/hosts` in UNIX systems and in `%systemroot%/system32/drivers/etc/hosts.txt` in Windows systems. In history, the `host.txt` was accessible via FTP for every peer but the problem was that the number of records grows and it did not scale well. Therefore, the proposal of RFC 882 and 883 in 1984 by Paul Mockapetris released a description of DNS with a hierarchical namespace. As the Internet grew, several documents were released and continued to developed DNS. The current specification of DNS is based on RFC 1034, 1035 and other specifications are published as a complement to implementation such as RFC 6891 describing extension mechanism for DNS or publication dealing with security issues such as DNSSEC [1].

### 2.1 Architecture

The DNS has become structured and namespace has increased because of the level approach that took place with hierarchy model. It ensures unique names that can be

reused under different domains such as mail.a.com or mail.b.com. The hierarchy model is organized in a tree data structure consisting of levels or zones as illustrated in Fig. 2.1. From top-down, it includes Root name servers, TLD (Top Level Domain) name servers and authoritative name servers (Second Level Domain).

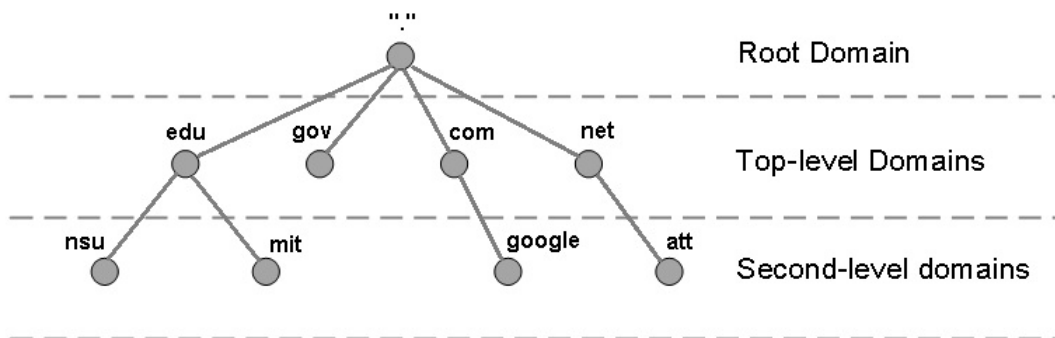


Figure 2.1: DNS Tree Structure [2]

### 2.1.1 Root Name Server

The root servers contain information about generic TLD (gTLD) such as .com, .net, .org and country code TLD (ccTLD) such as .cz for the Czech Republic. The rapid expansion of the Internet and need for domain name space cause creation of more TLD known as sponsored TLD (sTLD) such as .aero, .biz or .info. The root server is the top node of the tree and the authoritative server for TLD domain. It is recognized with "?" which is rarely displayed with FQDN. There are hundreds of running instances across continents which represent 13 root names servers. These servers are operated by 12 different organizations supervised by ICANN (Internet Corporation for Assigned Names and Numbers). The name convention follows X.root-servers.org with X representing an operating organization. The root servers are an essential part of the Internet and therefore the fault tolerance and availability is guaranteed by anycast routing.

### 2.1.2 TLD Name Server

TLD name servers are next in the hierarchy. These servers are operated by different organizations specified by IANA. Servers maintain information about second level domains that share common TLD. As an example, the .com gTLD contains information about ebay.com, facebook.com etc. The same rules applied to ccTLD which maintains national domains. CZ.NIC is responsible for .cz ccTLD in the Czech Republic and maintains

information about location of domains such as cvut.cz or seznam.cz.

### ■ 2.1.3 Authoritative Name Servers

Authoritative Name server is responsible for the list of subdomains that belongs to the specific domain. It contains detailed information about subdomain or third level domain such as www.google.com or ftp.google.com with corresponding records stored for different purposes (A, CNAME, MX etc. explained in section 3.1) [3].

## ■ 2.2 Resolution

The overview of resolution which stays in the background and takes care of name translation includes some parts of infrastructure not yet introduced such as recursive resolver and stub resolver.

### ■ 2.2.1 Recursive Resolver

A recursive resolver is a server responsible for finding recursively the response for a query in a domain hierarchy. The client assigns a task to recursive resolver to resolve a name for an IP address. The goal of a recursive resolvers is to provide name-address mapping for a client and cache queried data based on record TTL (Time-To-Live) to reduce number of later messages querying the same name. The information about recursive resolver is configured on a client either through DHCP option (Dynamic Host Configuration Protocol) or manual configuration. Resolvers are usually maintained by private network, organization, ISPs (Internet Service Provider) or CDNs (Content Delivery Network). The configuration of clients usually consists of two servers to implement high availability.

### ■ 2.2.2 Stub Resolver

The stub resolver is a component of a client computer that is accessed by an application program that desires name-address resolution. Stub resolver sends DNS queries and receives answers which are handed to an application program such as web browser or mail agent.

### ■ 2.2.3 Name Lookup

In Fig. 2.2, the client wants to visit a website <https://www.example.com> (step 1). To translate FQDN [www.example.com](https://www.example.com). to an IPv4 address, the client computer sends a DNS query for A record (name-IPv4 mapping) to recursive resolver (step 2). The

recursive resolver checks the local cache if the answer is already known. If the answer is unknown, resolver query root name server for a location of responsible .com ccTLD (step 3). The answer is received, and a resolver can subsequently query ccTLD for example.com (step 4). As a result, the resolver will get knowledge of location of the corresponding authoritative name server which controls a zone for domain example.com and lists a record for www.example.com. In the last step resolver contact name server responsible for domain example.com and receive a record including A record with IPv4 address and TTL which tells resolver how long the information should be kept cached (step 5). The answer is handed to client (step 6 and 7) and he can finally initiate a connection with the webserver on www.example.com. (step 8). The process seems to be time-consuming, but the lookups are achieved in milliseconds depending on cached data and distance of servers [4].

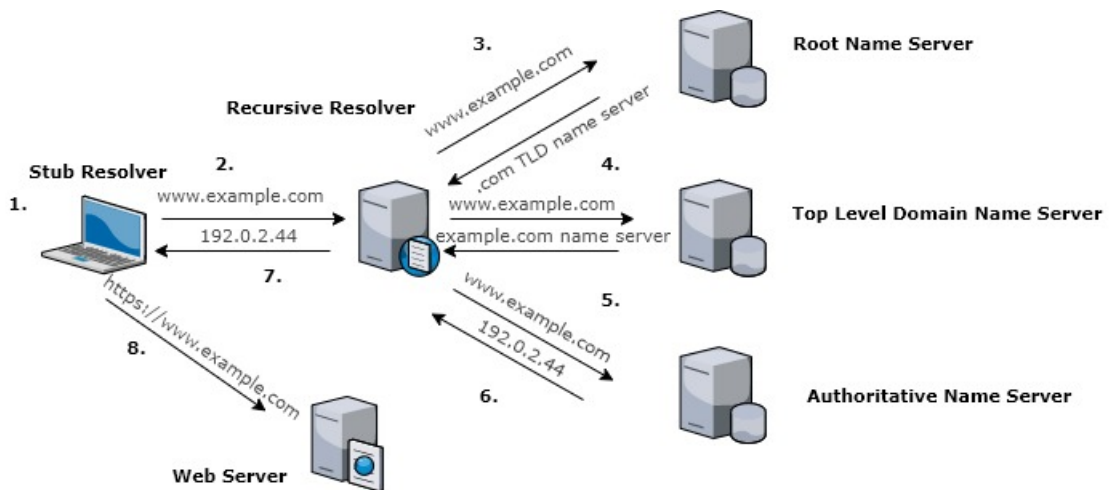


Figure 2.2: Name query for web page access [5]

## Chapter 3

### DNS Protocol

TCP/IP model presents DNS protocol in application layer above the transport layer which can be represented either by UDP (User Datagram Protocol) or TCP (Transport Control Protocol) both listening on port 53. The most usual is traditional connectionless UDP implementation with lower overhead and better performance. On the other hand, encrypted versions of DNS are exclusively using stateful TCP except for DNS over QUIC (Quick UDP Internet Connections) as a transport protocol which is more robust against attacks vector introduced later [6].

#### 3.1 Resource Record Type

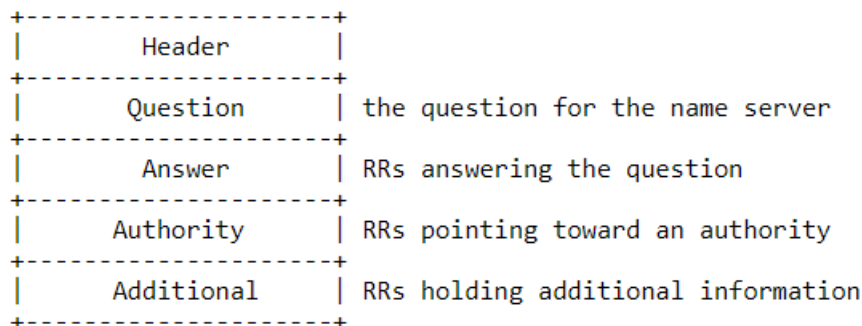
Before a description of a packet structure, it is necessary to introduce a Resource Record (RR) Type. The resolution between names and IPv4 addresses is a basic example of DNS functionality, although not an exhaustive explanation. To be more specific the illustrative example describes a DNS query for A RR type which result in the answer of IPv4 address. However, there are many types of RR that can be queried such as MX record mapping an email exchange server, AAAA record for IPv6 mapping, CNAME for an alias or SOA record which stores important information about a delegated zone (serial number, zone refresh time or an administrator). The most common RR types are listed in Tab. 3.1, every resource record is represented by corresponding value included in query or response message [7].

Value	RR Type	Description
1	A	Address record for IPv4 (32-bit IPv4 address)
2	NS	Name server; provides name of authoritative name server for zone
5	CNAME	Canonical name; maps one name to another (alias)
6	SOA	Start of authority; provides authoritative information for the zone
12	PTR	Pointer; provides address to (canonical) name mapping; used with in-addr.arpa and ip6.arpa domains for IPv4 and IPv6 reverse queries
15	MX	Mail exchanger; provides name of e-mail handling host for a domain
16	TXT	Text; provides a variety of information (e.g., used with SPF anti-spam scheme to identify authorized e-mail servers)
28	AAAA	Name server; Address record for IPv6 (128-bit IPv6 address)
41	OPT	Pseudo-RR; supports larger datagrams, labels, return codes in EDNS0
255	ANY	Request for all (any) records

**Table 3.1:** RRs table overview

## 3.2 Message Structure

DNS messages are responsible for domain name information exchange and follow the same convention independently of the under-layer encapsulation with different sections in use. DNS message includes five sections showed in Fig. 3.1: Header, Question, Answer, Authority and Additional. The header section specifies which following sections should be present and distinguished between queries, responses, and optional types. The question section describes the query. The last three sections contain RRs for an answer, pointing towards authority and holding optional additional information.



**Figure 3.1:** DNS message structure, RFC 1035 [8]



- **Header:** The most notable fields of a header section are following. A 16-bit identifier ID for matching queries and responses. QR type field represents whether the message is query (bit set to '0') or response (bit set to '1'). TR (TrunCation) specifies if the message exceeded maximum length of 512 bytes. RD (Recursive Desired) field is set in the query to desire the resolver to make a recursive query. RA (Recursive Available) is set in a response which reports that recursion is supported. Three bits denoted as Z are reserved for future use and now are occupied for DNSSEC extension. The last part of a header contains count of resource records in the query, answer, authority or additional section, these fields correspond to QDCOUNT, ANCOUNT, NSCOUNT and ARCOUNT.
- **Question:** The question section contains three parts: QNAME, QTYPE and QCLASS. QNAME stands for a query name that holds a name to query. QTYPE specify a type of query such as '0x0001' for A record, 0x000F for MX record or 0x0002 for NS (Name Server) record. QCLASS specify a class of a query which is usually set to 0x0001 as IN stands for INTERNET. Despite of IN QCLASS, there are some legacy classes used for purposes that were not intended for. CHAOS class is one of them. It is recognized by value of 0x0003 and can reveal server information including version, hostnames, etc. in popular DNS server implementation BIND.
- **Resource Record sections:** The last three sections used for response messages share the same structure with variable number of resource records. The structure is shown in Fig. 3.2 and includes NAME, TYPE, CLASS, TTL, RLENGTH and RDATA. NAME contains domain name that was queried (same as QNAME in the query). TYPE is a subset of QTYPE represented by code corresponding to resource record. CLASS specifies class of data in RDATA field. TTL gives the number of seconds that the record can be cached. TTL value must also consider a compromise between performance and consistency. As an example of importance, attackers can use very short TTL for Fast flux DNS technique which hides phishing and malware delivery sites. Trailing pieces of resource record sections include RLENGTH with the length of the RDATA field and RDATA with the data of the response itself [8].

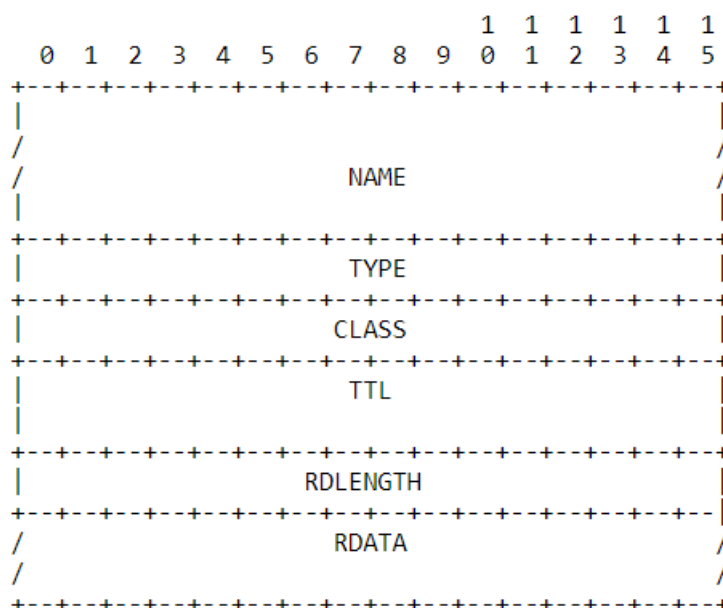


Figure 3.2: Resource Record format, RFC 1035 [8]

### 3.3 Extension Mechanisms for DNS (EDNS0)

DNS messages carried by underlying protocols must not exceed the maximum allowed length of 512 bytes (Transport and Internet headers are not included). Over time, it started to be a limiting factor for messages using IPv6 or security extensions. Therefore RFC 6891 introduces EDNS0 (Extension Mechanism for DNS) which extended functionalities of DNS specified earlier. The EDNS0 extension support is identified by an additional section with OPT pseudo-RR (Resource Record) recognized by decimal value of '41' in field TYPE. The RDATA of OPT pseudo-RR contains three parts illustrated in Fig. 3.3: OPTION-CODE which distinguish between each extension, OPTION-DATA holding data of extension and OPTION-LENGTH that report the size in octets of OPTION-DATA. The maximum payload of EDNS0 was increased to 4096 bytes [9].

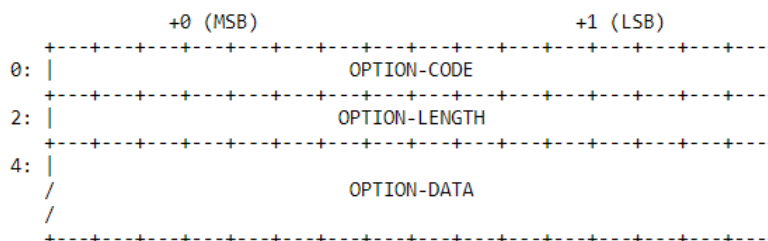


Figure 3.3: EDNS(0) format, RFC 6891 [9]

### ■ 3.3.1 EDNS0 Padding

One of the extensions relevant with messages encrypted in transport with DoT or DoH is EDNS0 padding. Although both queries and responses are encrypted with DoH or DoT, it is still possible to use a size correlation analysis to identify the content of messages. Therefore, EDNS0 padding increase the entropy of messages to defend against these types of attacks. The padding option must be supported by a requestor and it is recognized with a decimal value of '12' of OPTION-CODE. The OPTION-LENGTH is the size in octets of padding. The maximum size is defined by Requestor's Payload Size field encoded in the RR Class Field. There are different strategies to pad message, recommended is block-length padding. Other strategies include maximum-length padding, random-length padding, or random-block-length padding [10].

### ■ 3.3.2 EDNS0 Client Subnet

An option of EDNS0 Client Subnet (ECS) allows recursive resolvers to carry host network address information to upstream authoritative name servers. It is an extension specified with a decimal value of '8' in OPTION-CODE. The information about subnet from which query was sent intends to speed up the data delivery from CDN and load balance the traffic. If subnet information is gathered by authoritative name servers, they return an address of the geographically closest server and improve the latency and loading time of their services. Prior to the solution, authoritative name servers can get only address of recursive resolver that was serving the client's query. The issue comes with the privacy concern. The client subnet extension allows sharing client public IP address with another party which can be correlated with user activities following the query. Some privacy lead providers as Cloudflare or Quad9 disable ECS by default but many open resolvers do not operate the same. In conclusion, there are open resolvers supporting ECS and revealing less client information using only ASN (Autonomous System Number) from which the request comes from to blend client information and get only the most decisive information for optimal reachability – geographic location and not more specific client identification [11].



## Chapter 4

# Security of DNS Architecture

Internet services highly depend on the DNS architecture which does not significantly change through the years. As one of the Internet pioneer Vint Cerf claimed, he fears its security very much because of service misuse or disruption could cause very serious global harm [12]. Therefore, considerations must be taken in mind such as availability of DNS services and protection against amplification and reflection attacks. The consequences of being DNS services non-operation are extensive. The following issues should be taken in mind:

- There were several attempts to disrupt DNS root servers and cause Internet outage, although without any success [13].
- Authentication of messages to protect against forged redirection is another problem to deal with. Several techniques enabled to poison resolver cache with fraud record to redirect connection to malicious site.
- The traditional implementation and plaintext format of DNS messages lead to observation by multiple actors. The surveillance can be conveyed to monitor user behavior or disrupt services by a political regime which significantly violates user privacy.
- Monitoring is also a very useful tool for security analysts in enterprise managed networks to protect against malicious queries or identify infected devices.

### 4.1 DNS Reflection and Amplification Attacks

To reflect and amplify volume of traffic, cybercriminals misuse open recursive resolvers and conduct DDoS (Distributed Denial of Service) attack on a victim target. They send queries from multiple machines to resolver with a spoofed address which belongs to a

victim. The intention is to overwhelm a victim with a massive traffic volume. The lookup request typically submits payload with maximum size (4096 bytes) which is possible so that EDNS0 is required or the resource record type ANY is queried. The responses coming from the legitimate open resolvers target the victim system and tries to disrupt running services. The defence against these types of attacks includes rate-limit number of queries or stateful inspection of queries and rejection of requests with spoofed addresses [14].

## ■ 4.2 DNS Flood Attack

The attackers attempt to flood DNS request from multiple infected devices and location to exhaust resolver computing resources and make services unavailable. The high volume of data is typically generated from botnets consist of many IoT devices to cause DDoS attack. Unlike reflection and amplification, the target for attackers becomes the provider of recursive resolver. As a result of a successful attack, the legitimate users cannot access DNS resolution and many services become unavailable for them. The availability of providers is usually protected by an anti-DDoS protection offered by third parties or anycast network which can mitigate consequences. Anycast network allows multiple geographically distributed servers to share the same IP address. The requests are load balanced and routed with BGP (Border Gateway Protocol) to multiple servers and therefore they mitigate resources exhaustion [15].

## ■ 4.3 DNS Spoofing

The spoofing attack was a serious problem because it forges records in the resolver cache and the client was redirected to a malicious site. Even worse the record is usually cached so if an attack was successful the forged record remains in the cache until TTL expires. The attack exploits weak message IDs implementation. So that attacker query recursive resolver to get number of message IDs and when the recursive resolver contacts authoritative name server, the attacker responded with correctly guessed ID and before the genuine response as illustrated in Fig. 4.1. As a result, the resolver is tricked to believe that a hostname is residing on incorrect malicious IP address. The attack was described by Kaminsky in 2008 and countermeasures include source port and IDs randomization to increase entropy and make it harder to guess correct ID and source port of a transaction. Later DNSSEC was introduced to reduce cache poisoning attacks by authenticating responses. DNSSEC is based on the chain of trust, the records for

a zone stored by authoritative name server are signed with private key using digital signatures. The zone public key is published to public and recursive resolvers will fetch it to validate authenticity of a record. In order to trust the public key, it is signed by parent zone so that the domain icann.org public key is signed by the .org TLD [16]. DNSSEC propose four new resource record types: Resource Record Signature (RRSIG), DNS Public key (DNSKEY), Delegation Signer (DS) and Next Secure (NSEC). There are also new messages headers bits in field Z (reserved for future): Checking Disabled (CD) and Authenticated Data (AD) [17].

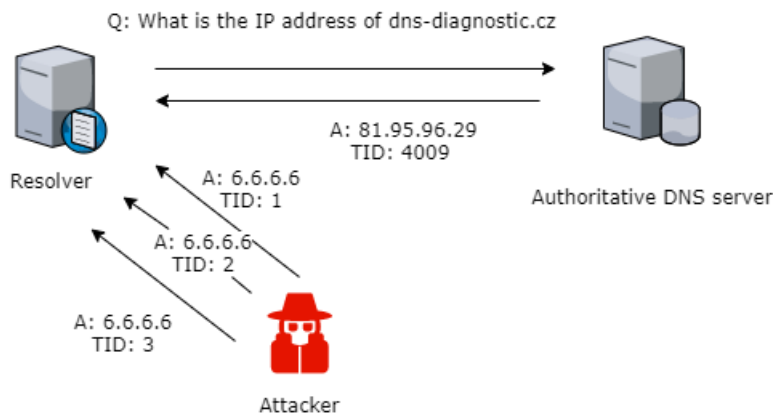


Figure 4.1: Spoofing Attack [18]

## 4.4 DNS Surveillance

DNS was not designed to address security and privacy in early development, therefore plaintext form of DNS messages is an issue nowadays regarding user privacy and possible inappropriate surveillance. The surveillance problem includes consumer activity commercialization and information access prevention required by the government. On the other hand, DNS traffic control has its importance in private network protection defending users against cybercriminals with security technologies such as enterprise firewalls, antivirus or parental control.

As DNS queries reveal information about identity through public IP address and user activity is recognized by DNS query, the data could be potentially harvested by ISP (Internet Service Provider). These data can be monetized and sold to companies which can better target advertisements for consumers. The monetization of user data differs across the world and most cases come from the USA (United States of America) ISP which can record, store, and sell data that passes through their servers [19]. These concerns include Verizon user “Supercookie” tracking case or ISPs hijacking which is





## Chapter 5

# Transport Layer Security Protocol

Transport Layer Security (TLS) protocol establishes a secure channel between two peers and protects communication over the Internet. It protects the overlaying application layer services such as HTTP (Hypertext Transfer Protocol), FTP (File transport protocol), IMAP (Internet Message Access Protocol), SMTP (Simple Mail Transfer Protocol) or DNS and it is built above the transport layer TCP or UDP. The description of TLS is an essential part for the purpose of DoH and DoT adopting TLS as an encryption layer encapsulating DNS queries. TLS was formerly known as Secure Socket Layer (SSL) protocol developed by Netscape Communications Corporation in 1994. Later, SSL came under IETF development and protocol framework was renamed to TLS. IETF continues development and following versions were released: TLS 1.0 (RFC 2246), TLS 1.1 (RFC 4346), TLS 1.2 (RFC 5246), and TLS 1.3 (RFC 8446). TLS 1.0 and 1.1 are beginning to be considered as deprecated versions nowadays, because of weak cryptography and vulnerabilities such as BEAST or POODLE [25].

As of February 4, 2021, SSL Pulse monitoring tool showed TLS 1.2 the most supported protocol (99 %) on the websites and TLS 1.0 and 1.1 is still supported almost on half of surveyed sites. On the other hand, TLS 1.3 released in 2018 is slightly increasing from previous year. The 41 % of websites supported the newest protocol version [26].

In the following chapter, TLS 1.2 and TLS 1.3 are described and compared. A closer look at cryptographic information essentially extracted from a handshake is examined.

## 5.1 TLS Protocol Overview

TLS protocol enables a secure connection between two peers over the transport layer protocol. To establish protected communication, authentication, confidentiality and integrity must be provided. Peers negotiate a protocol version, choose cipher suite which includes cryptographic algorithms for secure channel establishment and authenticate each

other (usually the server is authenticated only) using asymmetric cryptography. As an initial phase of exchange information is completed, peers derive a symmetric session key and establish an encrypted channel. In addition to address integrity, transmitted data are protected with message authentication code and cannot be modified by adversaries. The main building block of TLS is Record Protocol with additional record content types (subprotocols): the handshake protocol, the alert protocol, the change cipher spec protocol, and the application data protocol. The handshake protocol authenticates peers and negotiates protocol version, cryptographic algorithms, or cipher suites to establish symmetric session key. The change cipher spec protocol ensures encryption of subsequent messages during handshake. The application data protocol carries messages packaged, fragmented, and encrypted by record layer and the alert protocol carries notification with alert severity warning or fatal. The observation of handshake process reveals several parameters to evaluate and therefore it is discussed below.

## 5.2 Handshake Protocol

As TLS 1.2 is still a better-supported protocol and TLS 1.3 is influenced by previous version, the explanation of handshake would be based on TLS 1.2. The Fig. 5.1 shows each step of an initial channel establishment. The client always starts with the *ClientHello* message with its cipher suites, random bytes, the protocol version, session ID (empty if resumption is not in a place) and extension field. The server responds with *ServerHello* message with selected connection parameter and *Certificate* message with X.509 certificate. The *ServerKeyExchange* and *ClientKeyExchange* message hold additional information for key exchange based on decided cipher suite. Server then signals with *ServerHelloDone* that all messages were received. As client receives *ServerHelloDone* it forms *ChangeCipherSpec* message to tell the opposite side to start encryption, because all necessary information were already received. Therefore, the final *Finished* message completing the handshake process is encrypted and inside a payload it keeps verify data to ensure integrity and defence against man in the middle attack. The cipher suites negotiated during handshake influence further key exchange, authentication and integrity validation. The problem with TLS 1.2 is that some of the cipher suites use weak cryptography and are no longer safe to use [27].

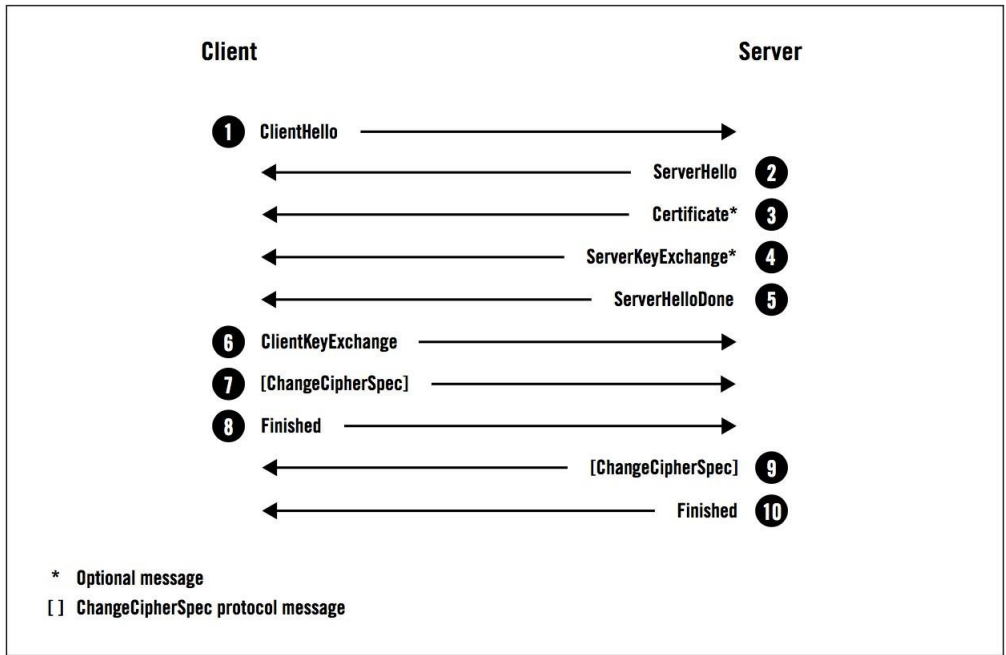


Figure 5.1: Client-Server handshake [27]

### 5.2.1 Key Exchange and Authentication

For key exchange, the RSA (Rivest-Shamir-Adleman) or authenticated Diffie Hellman (DH) is used. The RSA key exchange is not convenient, because it is missing the forward secrecy and there is a risk of a passive attack and later decryption if the server’s private key is leaked. DH based on difficulty in solving the discrete logarithm problem is much better choice, but it is missing the authentication. Thus, the combination of DH with RSA or ECDSA must be in place. To provide perfect forward secrecy DH is modified to use dynamic keys which is known as Diffie-Hellman Ephemeral (DHE). Diffie Hellman Ephemeral will always end up with a different premaster key. To make key exchange faster elliptic curve (EC) cryptography is combined with DHE which is known as ECDHE.

An illustration of DHE with RSA authentication in TLS 1.2 follows in Fig. 5.2. After the server chooses cipher suites offered by client, it generates a random private value  $b$  and proposes generator  $g$  and prime number  $p$ . The server takes the  $g, p$ , compute public value  $g^b \text{ mod } p$  and together with server random  $n_r$  and client random  $n_i$  it signed them with RSA private key  $sk_R$ . These values are sent together in *ServerKeyExchange* after *Certificate* message. The client verifies certificate validity and with extracted public key from certificate compares signed values and values received in a message. If values match,

the client computes  $g^a \bmod p$  and send it to server. As a result, the client and server can generate shared secret from  $g^{ab} \bmod p$  called as pre-master secret [28].

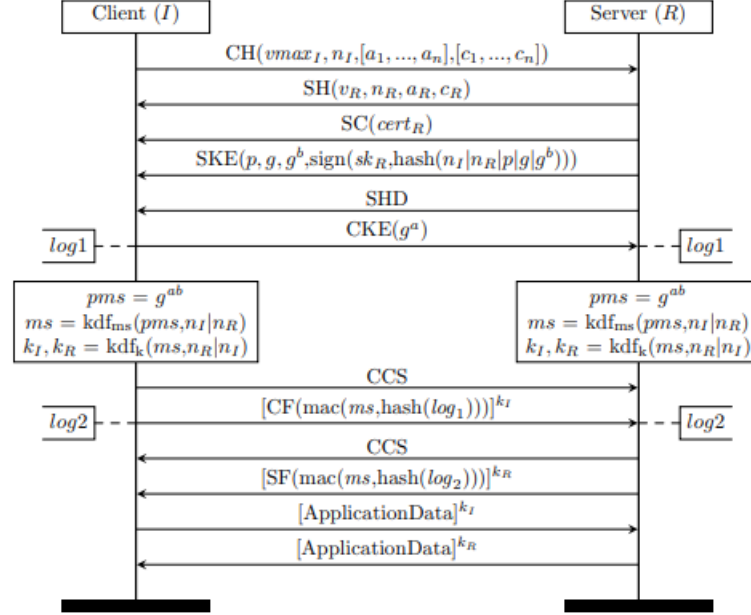


Figure 5.2: Message sequence for TLS 1.2 with (EC)DHE key-exchange [29]

The pre-master secret shared between each side is in TLS 1.2 used to create 48-byte master secret with PRF (pseudorandom function) using HMAC SHA256. The master secret then servers use as an input for session key generation ( $k_i, k_r$ ) for both direction which differs based on previous algorithm agreement [30]. The calculation follows the the formula:

$$\begin{aligned} master\_secret = PRF(pre\_master\_secret, "mastersecret", \\ ClientHello.random + ServerHello.random) \end{aligned} \quad (5.1)$$

### 5.2.2 X.509 Certificate

A certificate is a digital document that includes public key and identity information about associated entity signed with digital signatures to provide authenticity. The digital signature is a hash of a document encrypted with the private key of issuing certificate authority (CA). The current format X.509 version 3 is documented in RFC 5280. Certificates are issued by CA either root CA or intermediate CAs. To trust the certificate, the CA must be included in trusted store of device that is initiating a connection. There

are several fields of a certificate as version, serial number, signature algorithm, issuer, validity, subject and public key. The subject contains distinguished name (DN) of an entity associated with certificate and organization information. The common name (CN) inside subject reflects hostname. The similar information contains Subject Alternative Name (SAN) extension which can include multiple hostnames. The issuer part contains DN of certificate issuer. If dealing with self-signed certificates the issuer and subject fields are the same. The validity represents starting and ending date for certificate. The extensions introduced in version 3 extend some more important parameters such as SAN showed above, revocation control with CRL (Certificate Revocation List) Distribution Points or Authority Information Access field. CRL Distribution Points determine location of CRL through HTTP URI but in practice become ineffective, because the verifier had to search through the CRL in order to found revoked records. Therefore OCSP (Online Certificate Status Protocol) defined by HTTP URI in Authority Information Access is more common revocation control mechanism providing status information to the client. There also some new possibilities of certificate control such as Certificate Transparency [31]. The following Fig. 5.3 shows certificate information for *fel.cvut.cz* including issuer, subject, validity, public key information and extension fields. The output was gathered with openssl tool [32].

```

jan@jan-VirtualBox:~\$ openssl x509 -in fel_cvut_cz.pem -text -noout
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      04:fc:1c:60:5c:37:7c:84:a9:36:2b:17:57:65:4e:0a
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C = NL, ST = Noord-Holland, L = Amsterdam, O = TERENA, CN = TERENA SSL
↪ CA 3
    Validity
      Not Before: Dec  9 00:00:00 2019 GMT
      Not After  : Dec 13 12:00:00 2021 GMT
    Subject: C = CZ, L = Praha, O = Ceske vysoke uceni technicke v Praze, CN =
↪ www.fel.cvut.cz
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public-Key: (2048 bit)
      Modulus:
        00:dd:dc:da:d2:85:38:21:b8:34:de:66:c8:33:8e:
        c1:21:21:f4:7c:30:fd:bb:5d:47:50:58:63:61:8e:
        ....
      Exponent: 65537 (0x10001)
    X509v3 Subject Alternative Name:
      DNS: fel.cvut.cz, DNS: www.fel.cvut.cz, DNS: www.feld.cvut.cz
    Authority Information Access:
      OCSP - URI: http://ocsp.digicert.com
      CA Issuers - URI: http://cacerts.digicert.com/TERENASSLCA3.crt

```

**Figure 5.3:** Certificate verification with Openssl

## 5.3 Differences between TLS 1.2 and TLS 1.3

Major differences between TLS version 1.2 and 1.3 affect key exchange, handshake latency, less visibility of handshake initial state and omitting legacy cryptographic algorithms. TLS 1.3 moves more parameters into an extension field to ensure interoperability with middleboxes. Hence, the record protocol version use always values 0x0301 (TLS 1.0) or 0x0303 (TLS 1.2). TLS 1.3 (0x0304) support is moved to the extension field *Extension: supported\_versions*. The newer version specifies only five possible cipher suites and key exchange is easier to determine using (EC)DHE, PSK-only or PSK with (EC)DHE. TLS 1.3 is at least 1 round-trip faster and RSA and other static key exchange methods are

removed from TLS 1.3.

The intention is to start encryption as soon as possible and for that purpose, the Diffie-Hellman parameters are included in *ClientHello* message extension field *Extension: key\_share*. *ClientHello* message sends the list of cipher suites and guesses the key agreement (ECDHE with X25519 or P-256 curve) algorithm with a 'key share' value. As a result of successful agreement, *ServerHello* message derives shared secret and immediately can encrypt its initial messages including digital certificate which is not visible to public anymore. If the initial agreement fails, the *HelloRetryRequest* message is triggered.

Another improvement is a reduction in cipher suites. TLS 1.2 recommends 37 ciphers including weak algorithms as DES, MD5 or AES-CBC Mode. TLS 1.3 recommends only 5 cipher suites presented in Tab. 5.1 with AEAD (Authenticated Encryption with Associated Data) ciphers protecting data against modification during transmission. The weak cipher suites offered in TLS 1.2 were exploited by attackers who performed downgrade attacks. They were focused on an initial handshake between peers that was not completely protected. Accordingly, TLS 1.3 signed handshake message entirely and protect peers against mentioned downgrade attacks that could possibly occur [33].

AEAD Cipher Mode	HKDF Hash Algorithm
AES 128 GCM	SHA 256
AES 256 GCM	SHA 384
CHACHA20 POLY1305	SHA 256
AES 128 CCM	SHA 256
AES 128 CCM 8	SHA 256

**Table 5.1:** TLS 1.3 cipher suites





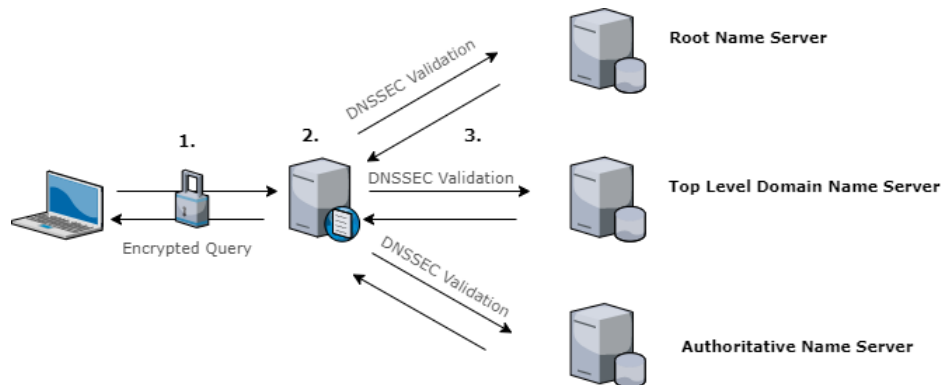
## Chapter 6

### Encrypted DNS

Over the years, open resolvers stopped being operated by ISPs only and began to be provided as public services (e.g. Cloudflare 1.1.1.1 or Google 8.8.8.8). As a cleartext packet had to travel through the Internet, the security issues arise. At first, defence against a message modification with authenticated DNSSEC between authoritative name servers and recursive resolver was introduced, but there was not an implementation to address authenticity and confidentiality between the client device and recursive resolvers. Exposure of user Internet activity and relatively easy capture of this cleartext information was a problem to deal with.

DNSCrypt was the first protocol developed to secure exposed client-resolver communication and it was implemented in 2011 by OpenDNS. Unfortunately, the protocol was not proposed to IETF and RFC was never published. Therefore, a broad adoption did not happen [34]. In 2016, DNS over TLS (DoT) was published in RFC 7858 and it was the first standardized solution to secure communication between clients and resolvers. DoT listen on dedicated port 853 and exchanged messages are secured with TLS. Two years later, DNS over HTTPS (DoH) was introduced in RFC 8484 secured with TLS layer as well but the structure of DNS message becomes a part of HTTP/2 protocol. DoH servers listen on port 443 and they are specified by URI template. Rather than traditional resolver assignment and handling queries by stub resolver, DoH began to be widely built-in inside a web browser application which was a significant change in conventional architecture [35]. The development of DoH is pushed by web browsers developers such as Mozilla Firefox and Google Chrome. To complete the portfolio of encrypted DNS, Google is working on DNS over QUIC, which seems to have promising performance but has not been standardized yet. The following chapters are focused on RFC standards DoH and DoT including related topics. In Fig. 6.1, it can be seen an illustration of encrypted DNS. The step 1 shows encrypted query that takes place between client and resolver only. The queries heading from a resolver are authenticated

if DNSSEC is in use (step 2 and step 3).

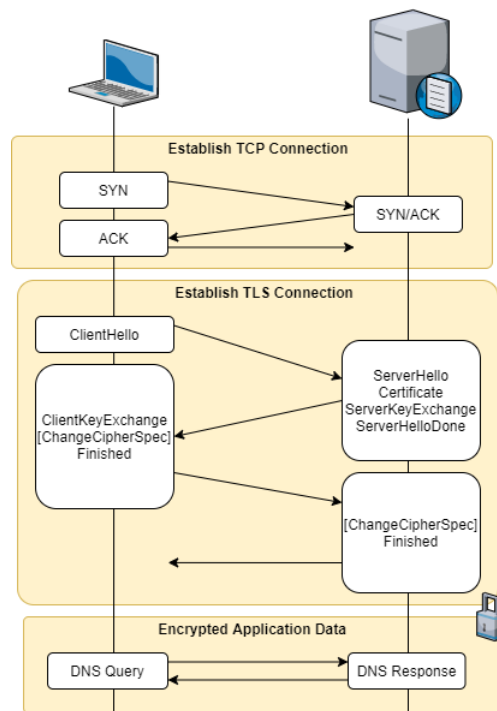


**Figure 6.1:** Encrypted DNS query message

## 6.1 DNS over TLS

To address privacy between client and resolver and encrypt DNS messages, DoT protocol was standardized by IETF in 2016. The protocol encapsulates DNS messages inside TLS to protect queries and responses from eavesdropping and hijacking. The structure of these messages follows RFC 1035 and remain the same as known from DNS over UDP [36]. DoT server accepts connection on port 853 reserved by IANA, therefore every DoT client must establish a TCP connection to port 853 unless it has an agreement with a server to use a different port. Upon a successful TCP connection, TLS handshake follows. After TLS establishment, the channel is secured against on-path attacker and DNS messages can be exchanged.

Security strongly depends on TLS and PKI (Public Key Infrastructure), the recommendation for TLS parameters follows RFC 7525. DoT defines two usage profiles: Strict Privacy Profile and Opportunistic Privacy Profile. The Strict Privacy Profile requires encryption and authentication of a DoT server. If these requirements are not satisfied, the failure occurs, and resolution is unavailable. On the other hand, with Opportunistic Privacy Profile availability is the most important and authentication of server is not required. The connection can result in encrypted communication with unauthenticated server or if both encryption and authentication fail, the fallback mechanism can switch to the cleartext communication [37]. DoT clients can learn of a TLS enabled DNS server from untrusted source such as router via option in DHCP message. Fig. 6.2 shows initial connection process which goes through TCP layer up to Application layer and data exchange.



**Figure 6.2:** DNS over TLS communication

## 6.2 DNS over HTTPS

In 2018, DNS over HTTPS (DoH) was standardized in RFC 8484 to protect against third party observers and to allow a web application to access DNS information through web browser API (Application Interface). DoH provides identical protection as DNS over TLS with some changes of encapsulating layer presented with HTTPS and DoH server listening on port 443. Every DNS message is transformed into HTTP/2 structure and protected by TLS. HTTP/2 is recommended version because of much better performance than earlier versions which should not be used. It also solves head-of-line blocking and supports reordering, parallelism, priority, and a header compression. Opposed to HTTP/1 which send replies in serial order and subsequent queries had to be put on hold if the previous query was delayed [38]. HTTP implementation also introduces new media formatting type *application/dns-message* intended for DNS messages.

DoH servers are configured via URI template, which constructs an URI for resolution. The URI definition is based on out of band configuration either through manual configuration or automatic assignment through DHCP option. The specification of URI is not provided, but many implementations are using path *\*/dns-query* or *\*/resolution*. Every DoH server must implement HTTP POST and HTTP GET methods. If the client

requires the resolution through HTTP POST, the DNS message is included as a body of request with header value Content-Type indicating media type *application/dns-message*. There is also a draft RFC for */application/dns-json* Content-Type supporting JSON format but is not mandatory for DoH servers [39]. HTTP GET method is processed through URI with variable *'dns'* and content of a DNS message encoded with base64url. The difference between two methods is that GET methods are better cached, but the size of a message is limited by URI size. The POST method transfers data inside the body and can have larger size. The recommendation for privacy-sensitive application is to use HTTP POST which increases latency and responses become hardly cacheable. It also gives protection against timing attacks from actors trying to determine the user activity visited lately [40]. Data exchanged through both methods follow DNS wire format specified in RFC 1035. Only modification of encapsulated traditional DNS wire format includes IDs which should be set to 0 in every request for HTTP correlation of requests and responses. Otherwise, with different IDs, the equivalent queries would be cached separately. The following Fig. 6.3 and Fig. 6.4 describe HTTP/2 GET and HTTP/2 POST method that carry encoded DNS message in different places.

```
:method = GET
:scheme = https
:authority = dnsserver.example.net
:path = /dns-query?dns=AAABAAABAAAAAAAAA3d3dwdleGFtcGx1A2NvbQAAAQAB
:accept = application/dns-message
```

**Figure 6.3:** DoH with HTTP/2 GET message [41]

```
:method = POST
:scheme = https
:authority = dnsserver.example.net
:path = /dns-query
:accept = application/dns-message
:content-type = application/dns-message
:content-length = 33

<33 bytes represented by the following hex encoding>
00 00 01 00 00 01 00 00 00 00 00 00 03 77 77 77
07 65 78 61 6d 70 6c 65 03 63 6f 6d 00 00 01 00
01
```

**Figure 6.4:** DoH with HTTP/2 POST message [41]

## 6.3 Privacy Consideration

Almost the same equivalent of privacy gives both types of encrypted DNS against observer who might conduct user behaviour analysis or control information access. However, these general terms should be discussed in more detail regarding the situation in the world. In the USA, ISPs can gather DNS messages to big data analytics and monetize these data. In the UK or China, ISPs are obliged to monitor DNS traffic to stop unlawful activities (the specification of unlawful activities differ in these countries) and European countries prevent data gathering by ISPs with GDPR. In conclusion, DoT and DoH can have different implication in these different situations and in some countries enabling DoT and DoH can have more impact than in the others.

### 6.3.1 Client-Query Mapping

There are also some privacy concerns with DoH and DoT. In traditional DNS, resolvers have visibility of a public address of received client query and they can expose subnet information through ECS to authoritative servers. The same information leakage applies to encrypted DNS and the new privacy concern arising on a client-resolver path only. DoT and DoH are stateful protocols and for better performance, multiple queries should be exchanged through a single session. As a result, a single session can expose multiple queries mapped to public address and a source port. Traditional DNS over UDP exchanges queries with different source ports, therefore the detailed client-query mapping was not possible from the resolver operator point of view.

The DoH can become even more specific to the client-activity matching. DNS over UDP was limited to identity by public address which hides several users behind NAT (Network Address Translation) [42]. Nevertheless, DoH introduces new concerns with HTTP layer and its header including user-specific information like Cookie, User-Language and Accept-Language which could be misused. Together with long-lived session for performance purposes and multiplexed queries through it, the end user can reveal more information to DoH server. Although the protocol designers discourage providers to use HTTPS cookies to prevent tracking [43], the RFC 8484 does mention the consideration of HTTP cookies, but states that cookies should not be accepted by DoH clients unless they are explicitly required by a use case [35]. The trustworthiness of DoH was even doubted when Mozilla Firefox introduced plans to enable DoH in the USA by default. Nowadays, the DoH is enabled by default only in the USA however, if the managed network/parental control is discovered through the feature called canary domain, the DoH is disabled. Other

countries can optionally enable DoH in their web browsers [44]. In a conclusion, more user details could be potentially gathered with encrypted DNS because of its protocol design, therefore it is important to choose a trustworthy provider following personal data protection agreement and not exploiting information that would be present within encrypted messages.

## 6.4 Security Issues

Both encrypted DNS protocols give clients the same level of protection in terms of confidentiality. Messages are authenticated with underlying TLS depending on PKI, confidential and cannot be modified during transmission. Security issues are mostly affecting TLS protocol such as man-in-the-middle attacks, weak algorithms, downgrades attacks or metadata leaks. There are also some challenges arising including false sense of security and bypassing of cybersecurity defence measures. It is worth mentioning that the encryption occurs only in the ‘last mile’ between client a recursive resolver, the rest of the communication can be authenticated with DNSSEC only. Clients should also be informed that DoT and DoH do not completely hide the user activity. There are still pieces of data leaking from other layers of communication such as SNI (Server Name Indicator) in TLS header for desired host recognition and IP address of service for routing purposes. As conclusion, it is questionable whether DoH and DoT can completely stop surveillance. It can be rather said that surveillance can be much more difficult to perform [45].

### 6.4.1 Correlation Analysis

Even though a secure channel is established, there is still a possibility for traffic analysis of encrypted data based on metadata such as timing and size correlation. To introduce the problem, the overhead of protocols remains the same, but a query for different domain can differ in length. Therefore, size information can reveal content of a message. The protection includes padding which makes content identification more difficult and can be added through EDNS0 padding or in HTTP through HTTP/2 padding. Optional EDNS0 padding policies can be implemented in both encrypted versions through application layer of a DNS message to defend encrypted queries against size correlation for the cost of increased volume of data [46]. Based on empirical research, recommended settings for clients are to pad queries to the nearest of 128 bytes. As a response to such message, DoT enabled resolvers pad responses to the nearest multiple of 468 bytes [47]. However, as the result shows from researchers, padding cannot completely stop DNS fingerprinting and

traffic analysis. Using Random Forrest classifier, analysis is still possible even if padding is used, but increased size of padding slightly makes the correlation attacks harder [48].

### 6.4.2 Encrypted DNS in Enterprise Environment

DoH and DoT have their importance for individuals at SOHO (Small Office Home Office) networks but the misuse of these in enterprise-controlled networks can have severe consequences. There is a serious risk of using encrypted DNS with resolver not managed or not allowed by organization because it can establish an exfiltration channel and pass through malware or commands and controls messages. Therefore, identification of potentially infected device is not possible. The DoT communication can be easily recognizable with dedicated port 853. Much more exhaustive investigation is needed with DoH which hides together with HTTPS traffic targeting port 443. Thus, become a useful tool for an attacker. The attacker can use the same technique as user does to protect their privacy with DoH and hide its activity against observer which is in this case a network administrator. There is a proof of concept implementation such as GODOH showing the misuse of DoH is possible [49]. In 2020, the first attempt of the Iranian hacking group was made to incorporate DoH in its attacks [50]. If DoH becomes more popular it is very likely that more attacks would be introduced. The detection of DoH misuse hiding with regular web browsing traffic would be expensive for managed organization networks because full proxy solution or machine learning models would be necessary to recognize malicious traffic. On the other hand, the study confirms that identification of clients using DoH is possible by ML algorithm Ada-Boosted Decision tree [51]. Consider a sufficient likelihood to identify this type of traffic only on metadata information gives security analysts chance to put an on-path device (proxy) and interfere content of exchanged messages. As conclusion, there are continuous debates about impacts of DoH and some of the security professionals discourage public from a wide adoption [52].

## 6.5 Performance of Encrypted DNS

In comparison between traditional and encrypted DNS, there is an increased cost for privacy extension. An overhead and latency caused by TCP connection establishment and TLS layer become a challenge to deal with. Therefore, several techniques were introduced to minimize latency. At first, clients should combine multiple queries over a TLS session without waiting for a response. The cost for TCP and TLS connection setup is relatively high, the clients should not immediately close the connection after the answer is received and the connection should be reused for forthcoming queries and

become idle for some time if necessary. It is a recommendation for client performance as well as protection for resolvers against resource exhaustion and denial of service (DoS). The RFC does not specify the timeout for client or server, and it depends on available resources during high (shorter timeout) or low activity (longer timeout) period.

TCP Fast Open (TFO) defined in RFC 7413 is another performance enhancement for TCP improving reestablishment of connection on a transport layer. The efficiency is based on the key fact that data can be delivered to an application before the TCP 3-way is completed. Client starts with TCP SYN including TFO cookie request option in a header. Server generates a cookie, which is a message authentication code (MAC) tag to be specific and it is unique to the client. Server sends back TCP SYN-ACK with TFO option including generated cookie and the client caches the cookie for later connection [53]. As a result, during the future connection establishment with a server, client will start with TCP SYN including cookie in TFO option altogether with data for application. The server verifies the cookie and if it is valid, TCP SYN-ACK is sent back to acknowledge SYN and the data from client. Otherwise, the server drops data for application and only TCP SYN is acknowledged. Despite the benefits TFO provides, it lacks widespread adoption. Middleboxes such as firewalls and proxies cause issues with unknown option TFO introduces. There are also tracking concerns and other performance improvements such as more popular HTTP/2 included with DoH and web services in general [54]. As related work discovers, only less than 2 % of observed open resolvers based on TCP supported TFO in 2019 [55].

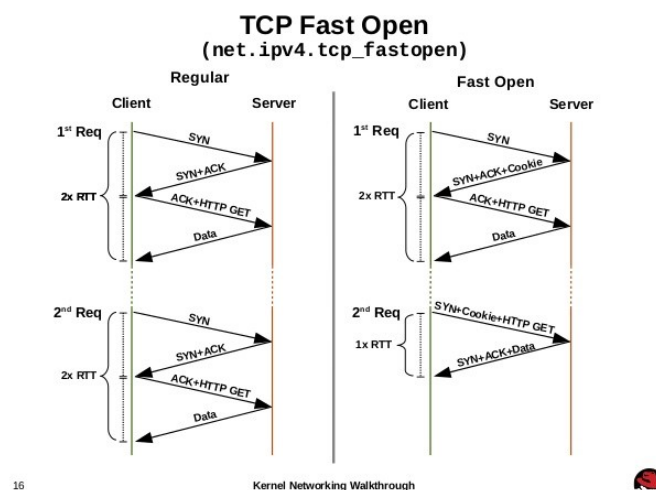
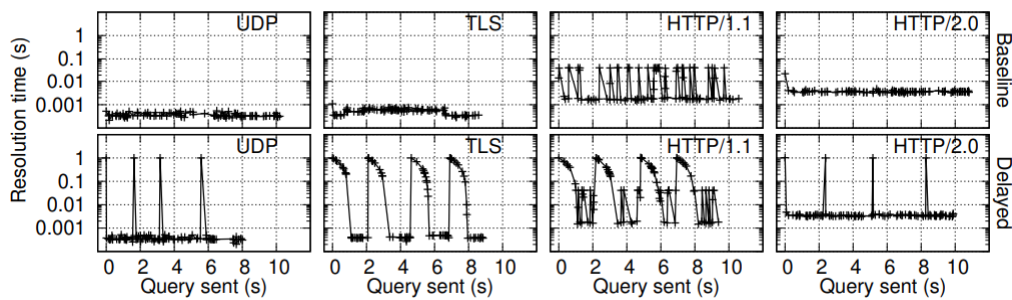


Figure 6.5: TCP Fast Open [56]



The next performance-related improvement includes 0-RTT (0-Round Trip Time) mode introduced in TLS version 1.3, which resume the subsequent session with minimum latency. It is built on PSK authentication and *early\_data* extension but the security of exchanged messages is downgraded because of possible replay attacks and loss of forward secrecy. Despite benefits of which 0-RTT offer, it lacks implementation on public recursive resolvers as well [46].

To compare the actual data, the performance of encrypted DNS resolvers was measured by an empirical study of researchers in 2019 [47]. The resolution time for DoT was less than one millisecond similar to UDP implementation and it has better capabilities than DoH. However, with introduced delay by resolver with every 25th query, the subsequent queries were impacted and delayed much more than UDP and DoH, because of missing implementation of out-of-order delivery by DoT servers. The Fig. 6.6 shows increased delay affecting TLS and HTTP/1.1 (not recommended with DoH). On the other hand, the number of measured providers was minimal not reflecting number of DoT providers in 2021.



**Figure 6.6:** Performance of encrypted DNS [47]





## **Part II**

### **Practical Part**



## Chapter 7

# Methodology of Encrypted DNS Data Collection and Analysis

The following chapter describes the procedure for detection of publicly available DoT and DoH servers providing recursive resolution. It additionally outlines an analysis of identified servers based on security and privacy properties.

The discovery of available resolvers is the first necessary step to take to get up-to-date information and differ in approach. DoT resolver has dedicated port number that is exclusively reserved and an open port scanning would be the most convenient way to reflect sufficient number of providers. To discover DoH resolvers, there are only few matching criteria to follow. The common port number 443 is not an appropriate option because of its usage in web services. Therefore, URI template identification with common part of *'/dns-query'* is chosen. Collected data are filtered afterwards to choose only legitimate implementations. These systems are further evaluated in the context of multiple security parameters included in DNS protocol and TLS layer with particular emphasis on the privacy enhancement protecting against sophisticated attacks, cryptographic algorithms and client identification. The whole procedure can be divided into:

- **Data Collection:** DoT resolvers are discovered by Internet-wide scan. As DoH resolvers share common port 443, data collection consist of database lookup of possible DoH resolvers from publicly available sources. In both cases, only resolvers which respond to DNS queries will be further assessed.
- **Analysis:** Observed data are analysed in the context of confidentiality, integrity, authenticity and privacy. From the technical point of view, an observation would be split up into followings layers of interests.
  - DNS Layer: It covers privacy extension EDNS0 padding, DNSSEC support by

resolver, ECS handling and information revealing software version or identity of a provider.

- TLS Layer: It includes further assessment of version, ciphers suites and certificates presented in handshakes.

## 7.1 Data Collection

DoT resolvers listen to incoming connection on predefined port 853 reserved by IANA and therefore identification of publicly available implementations was carried out by Internet-wide scan in IPv4 address space. The chance of missing DoT resolvers was put to a minimum. Omitted resolvers can include those provided by local ISPs, thus not available to public and those using a different port number which is unlikely to happen, because of reconfiguration of client settings would be necessary.

The measurement was performed with a convenient utility for that purpose Zmap (version 2.1.1) which is a fast, single packet network scanner designed for Internet-wide network surveys [57]. Zmap uses TCP SYN to discover open ports on 853. If an IPv4 address responds with TCP SYN/ACK, Zmap will mark the IPv4 address as reachable and add it to the list of *candidates*. The following command in Fig. 7.1 reflects analysis for IPv4 address space with destination port 853, bandwidth set to 20 Mbit/s, output of *candidates* to a file `zmap_dot_candidates.csv` and logs collection in `zmap_dot.log` during measurement. The bandwidth could be increased, but during pretests, higher speeds were causing insufficient memory space for buffering on a virtual machine Debian GNU/Linux 10 (Buster) with 8 vCPU and 6 GB memory. Accordingly, the selected values reflect the tradeoff between time and accuracy.

```
jan@jan-VirtualBox:~\$ zmap -p 853 -B 20M -i ens3 0.0.0.0/0 -o
↪ zmap_dot_candidates.csv &> zmap_dot.log
```

**Figure 7.1:** Internet-wide scan for port 853

DoH discovery is a more challenging task. According to the previous research [58], URI template identification of DoH resolvers and regular expression query for URI containing `/dns-query` was the most successful method to detect providers offering DoH services for public. These values are not specified by RFC, but implemented by most of the providers and almost become a naming convention. Database of URLs is necessary for this type of identification. These are usually stored by Threat Intelligence agencies providing

datasets of URLs for retrospective analysis of malware and collection of evidence for security incidents. For obvious reasons, this is a paid service and not available for free use. For that reason, DoH resolvers were collected from two sources. As the most reliable online source was selected one with comprehensive and frequently maintained data of URL published on cURL (Client for URL) github page [59]. Not to be dependent on this list only, it is considered that providers of DNS would offer its services for both encrypted DNS protocols, thus Internet-wide scan could be reused. The collected list of legitimate implementation of DoT servers provides certificates that contain Subject Alternative Name in an extension. This extension has information about DNS records connected with an IPv4 address. If DNS record is connected with */dns-query*, it creates a potential URL for DoH server. Accordingly, it is possible to find potential DoH servers listening on port 443 with this approach.

## 7.2 Analysis of DNS Layer

At first, discovered servers need to be filtered to reflect only legitimate implementations which provide encrypted DNS with recursion capabilities. The analysis can be performed afterwards. At the time of writing, the most convenient tool for analysis of encrypted DNS was KDIG (Knot Domain Information Groper) in version 3.0.5, an advanced DNS lookup utility usually bundled with an open source Knot DNS server developed by CZ.NIC [60]. It gives a variety of diagnostic options for a DNS layer including message header fields, question section, extension mechanism for DNS, latency and also supports an encryption layer through TLS or HTTPS. For the purpose of analysis it was registered a domain name *arnold.diagnostic-dns.cz* under .cz TLD with DNSSEC support, A record and TXT record providing information about measurement.

### 7.2.1 Initial Extraction and EDNS0 Parameters

After data collection of candidates for DoT and DoH, filtration takes place. DNS lookup for registered A record of *arnold.diagnostic-dns.cz* is performed. For DoH request HTTP/2 POST message is used. If a response is received, candidate is declared as DoT or DoH enabled server. The reflection of recursive resolvers is achieved so that the query is sent with field RD set to '1' in a message header. If recursive queries are supported, the server responds with RA bit set to '1'. As a conclusion, RD and RA field determines whether the server should be considered as recursive and queries should be further processed or discovered server will not be assessed anymore. It is also necessary to control the reply code flag in a message header. If flag gives different values than

NOERROR, servers are confirmed to support encryption but their status is recognized as not functional.

After initial data extraction, the DNSSEC and EDNS0 padding is reviewed. Investigated resolvers can optionally validate records from authoritative name servers. DNSSEC validation provided by recursive resolver is announced with AD (Authentic Data) bit set in a response header. It indicates that data in the answer and authority section were authenticated. It is also important to mention that the authoritative name server responsible for a certain domain does not have to undertake DNSSEC validation. Therefore, the investigation must be performed to query a domain involved in a chain of trust, otherwise, the resolver DNSSEC support cannot be validated. When validating the resolver support, the stub resolver has an option to include DO (DNSSEC OK) bit and receive a DNSSEC Records such as signature RRSIG (RRset Signature). However more convenient would be setting AD flag in a query (originally exclusive for responses, but later was redefined) as stated in RFC 6840 to get information about validation result without any additional DNSSEC records received [61].

Every resolver found by previous discovery is also verified whether EDNS0 padding is supported and how it is implemented. As experimentation RFC 8467 declared, the empirical research recommends block length padding with queries to a multiple of 128 octets and responses to a multiple of 468 octets. These values provide valuable protection against correlation attacks and take into account compromise between sufficient protection and acceptable overhead. KDIG utility uses with DoT query sensible amount of padding by default. The query also indicates EDNS0 padding option, therefore the response from resolver should contain at least some amount of padding. If it is not the case, implementation does not follow recommended settings. The following Fig. 7.2 shows query *anorld.dns-diagnostic.cz* for A record (default option) through DoT with padding, RD, AD flag set and timeout of 2 seconds. The response message contains RA, AD flags indicating recursive server, DNSSEC support and status NOERROR. The lower section contains line with received bytes which follow recommendation of 468 bytes of padding.



```

jan@jan-VirtualBox:~$ kdig arnold.dns-diagnostic.cz @1.1.1.1 +tls +timeout=2 +rdflag
↔ +adflag
;; TLS session (TLS1.3)-(ECDHE-X25519)-(ECDSA-SECP256R1-SHA256)-(AES-256-GCM)
;; ->>HEADER<<- opcode: QUERY; status: NOERROR; id: 1879
;; Flags: qr rd ra ad; QUERY: 1; ANSWER: 1; AUTHORITY: 0; ADDITIONAL: 1

;; EDNS PSEUDOSECTION:
;; Version: 0; flags: ; UDP size: 1232 B; ext-rcode: NOERROR
;; PADDING: 395 B

;; QUESTION SECTION:
;; arnold.dns-diagnostic.cz.                IN      A

;; ANSWER SECTION:
arnold.dns-diagnostic.cz.      2816    IN      A      81.95.96.29

;; Received 468 B
;; Time 2021-05-08 10:07:12 CEST
;; From 1.1.1.10853(TCP) in 27.9 ms

```

**Figure 7.2:** DNS query over TLS with required recursion and DNSSEC

To determine if the recursive server supports ECS Extension, it is possible to query the TXT record of `edns-client-sub.net` [62]. The authoritative server of `edns-client-sub.net` fetches information of ECS in a query from investigated resolver and result data in JSON. The JSON data are provided as a response for TXT record with TTL 0 to produce different output for each server. The response of a TXT query outputs whether EDNS Client Subnet is supported and which resolver was handling the query to the authoritative name server. In Fig. 7.3 the response contains the client subnet payload revealing the subnet information from which the query was received.

```

jan@jan-VirtualBox:~$ kdig edns-client-sub.net TXT @8.8.8.8 +tls
;; TLS session (TLS1.3)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)
;; ->HEADER<<- opcode: QUERY; status: NOERROR; id: 33381
;; Flags: qr rd ra; QUERY: 1; ANSWER: 1; AUTHORITY: 0; ADDITIONAL: 1

;; EDNS PSEUDOSECTION:
;; Version: 0; flags: ; UDP size: 512 B; ext-rcode: NOERROR
;; PADDING: 204 B

;; QUESTION SECTION:
;; edns-client-sub.net.                IN      TXT

;; ANSWER SECTION:
edns-client-sub.net.      0      IN      TXT      '{"ecs_payload':{'family':'j
↪ 1','optcode':'0x08','cc':'CZ','ip':'X.X.X.X','mask':'24','scope':'0}','ecs':'True'
↪  ','ts':'1620473374.15','recursive':{'cc':'US','srcip':'172.253.225.33','sport':'
↪  43507'}}'"

;; Received 468 B
;; Time 2021-05-08 13:29:28 CEST
;; From 8.8.8.8@853(TCP) in 291.5 ms

```

**Figure 7.3:** DNS over TLS query with ECS

## 7.2.2 Server Fingerprinting

Encrypted DNS does not modify the structure of DNS messages defined in RFC 1035, therefore secure implementations share some properties as traditional DNS. One of these is additional information that can be gathered from resolver to determine software version, hostname, authors or identification. This information can be abused by cybercriminals to carry out attacks against unpatched or deprecated versions of software. Therefore investigation of revealed information by DoT and DoH resolvers is an interesting section to have a look on because servers are open to public and could be a target of malicious activities. The query part of a DNS message contains QCLASS values which are very likely set to 'IN' (INTERNET) in traditional queries as described in the theoretical part. However, there are other legacy classes and one of them contains the value 'CH' as CHAOS (which was a network implementation that did not succeed). The class is well recognizable and "misused" by software distributors of DNS servers such as ISC BIND, Microsoft DNS Server or dnsmasq and it provides identification

which is configured by system administrator or left with default values. If the query is sent for a specific name with a resource type TXT and class CHAOS, the configured resolver will respond with information based on the query name. The query name can contain: version.bind, hostname.bind, authors.bind and id.server. The most common are version.bind and hostname.bind that can reveal information about software version and hostname. Additional information as authors.bind and id.servers are not as common but information about administrator or identification of machine can be gathered as well [63]. An example of a DoH query for software version is specified in Fig 7.4 with resulting software PowerDNS Recursor.

```

jan@jan-VirtualBox:~$ kdig version.bind TXT CH @XXXX.XX +https
;; TLS session (TLS1.3)-(ECDHE-SECP256R1)-(ECDSA-SECP384R1-SHA384)-(CHACHA20-POLY1305)
;; HTTP session (HTTP/2-POST)-(XXX.XX/dns-query)-(status: 200)
;; ->>HEADER<<- opcode: QUERY; status: NOERROR; id: 0
;; Flags: qr rd ra; QUERY: 1; ANSWER: 1; AUTHORITY: 0; ADDITIONAL: 1

;; EDNS PSEUDOSECTION:
;; Version: 0; flags: ; UDP size: 512 B; ext-rcode: NOERROR

;; QUESTION SECTION:
;; version.bind.                CH          TXT

;; ANSWER SECTION:
version.bind.                86400      CH          TXT          "PowerDNS Recursor 4.4.3
↔ (built Mar 30 2021 06:10:26 by root@b1a752bd07df)"

;; Received 127 B
;; Time 2021-05-08 17:07:13 CEST
;; From XXXX@443(TCP) in 55.4 ms

```

**Figure 7.4:** CHAOS Class and server fingerprinting

## 7.3 Analysis of TLS Layer

Most of the analysis is performed with KDIG which provide a sufficient amount of information about negotiated version of TLS and its parameters with a server. KDIG is able to negotiate session with TLS 1.0 to TLS 1.3 enabled server. To reflect specific cipher suites, groups for elliptic curves and signature algorithms which are offered by KDIG, the *ClientHello* message had to be reviewed. Thus, an initial message was captured by

Wireshark software and list of offered cipher suites (29 suites) is shown in Fig. 7.5. The list also contains weak cipher suites such as those not supporting forward secrecy. For the purpose of the surveys, it is desirable property because agreement on the weak cipher would mean that the server is not providing sufficient protection.

```

> Frame 25944: 391 bytes on wire (3128 bits), 391 bytes captured (3128 bits) on interface 0
> Ethernet II, Src: IntelCor_36:6d:a7 (94:65:9c:36:6d:a7), Dst: Technico_id:bf:40 (58:23:8c:1d:bf:40)
> Internet Protocol Version 4, Src: 192.168.0.134, Dst: 1.1.1.1
> Transmission Control Protocol, Src Port: 57813, Dst Port: 853, Seq: 1, Ack: 1, Len: 337
▼ Transport Layer Security
  ▼ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 332
  ▼ Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 328
    Version: TLS 1.2 (0x0303)
    Random: ef567021dff314689d871ed48ce4bca2cf0a5e92aa8259c2...
    Session ID Length: 0
    Cipher Suites Length: 58
  ▼ Cipher Suites (29 suites)
    Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
    Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
    Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
    Cipher Suite: TLS_AES_128_CCM_SHA256 (0x1304)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xc03a)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CCM (0xc0ad)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CCM (0xc0ac)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
    Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xc038)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
    Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
    Cipher Suite: TLS_RSA_WITH_AES_256_CCM (0xc09d)
    Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
    Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
    Cipher Suite: TLS_RSA_WITH_AES_128_CCM (0xc09c)
    Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x009f)
    Cipher Suite: TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xc03a)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CCM (0xc09f)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x009e)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CCM (0xc09e)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)

```

**Figure 7.5:** List of cipher suites offered by KDIG

From the *ClientHello* message could also be extracted group of elliptic curves (10 groups) found in *extension: supported groups* together with provided signature algorithms (16 algorithms) as illustrated in Fig. 7.6. If offered algorithms do not match on client and server, the communication is disrupted and session is not initiated. It would be unlikely to happen, because it is expected that server with encrypted DNS should support secure algorithms (not legacy ones) which are covered in offered ciphers.

```

  ▾ Extension: supported_groups (len=20)
    Type: supported_groups (10)
    Length: 20
    Supported Groups List Length: 18
    ▾ Supported Groups (9 groups)
      Supported Group: secp256r1 (0x0017)
      Supported Group: secp384r1 (0x0018)
      Supported Group: secp521r1 (0x0019)
      Supported Group: x25519 (0x001d)
      Supported Group: ffdhe2048 (0x0100)
      Supported Group: ffdhe3072 (0x0101)
      Supported Group: ffdhe4096 (0x0102)
      Supported Group: ffdhe6144 (0x0103)
      Supported Group: ffdhe8192 (0x0104)
  > Extension: ec_point_formats (len=2)
  ▾ Extension: signature_algorithms (len=32)
    Type: signature_algorithms (13)
    Length: 32
    Signature Hash Algorithms Length: 30
    ▾ Signature Hash Algorithms (15 algorithms)
      > Signature Algorithm: rsa_pkcs1_sha256 (0x0401)
      > Signature Algorithm: rsa_pss_pss_sha256 (0x0809)
      > Signature Algorithm: rsa_pss_rsae_sha256 (0x0804)
      > Signature Algorithm: ecdsa_secp256r1_sha256 (0x0403)
      > Signature Algorithm: ed25519 (0x0807)
      > Signature Algorithm: rsa_pkcs1_sha384 (0x0501)
      > Signature Algorithm: rsa_pss_pss_sha384 (0x080a)
      > Signature Algorithm: rsa_pss_rsae_sha384 (0x0805)
      > Signature Algorithm: ecdsa_secp384r1_sha384 (0x0503)
      > Signature Algorithm: rsa_pkcs1_sha512 (0x0601)
      > Signature Algorithm: rsa_pss_pss_sha512 (0x080b)
      > Signature Algorithm: rsa_pss_rsae_sha512 (0x0806)
      > Signature Algorithm: ecdsa_secp521r1_sha512 (0x0603)
      > Signature Algorithm: rsa_pkcs1_sha1 (0x0201)
      > Signature Algorithm: ecdsa_sha1 (0x0203)

```

**Figure 7.6:** List of elliptic curves and signature algorithms

Since encrypted version also depends on a PKI, information about providers, issuers and validity checks can be performed with OpenSSL (version 1.1.1d). The default configuration of OpenSSL is set to use only algorithms from security level 2 (minimum of 112 bits of security). To reveal resolvers with legacy ciphers support, level has to be lowered to level 1 (minimum of 80 bits of security) [64]. Certificates could be analysed subsequently by viewing the whole certificate chain or signature algorithms. For an analysis, only the most important parts are extracted. The subject field gives information of provider of encrypted DNS service and issuer gives CA provider signing the certificate. These values are helpful to recognize multiple servers and group them under one operator. OpenSSL also controls validity of certificates and reflect invalidated certificates. If information about OCSP is obtained, a certificate revocation check is made as well.



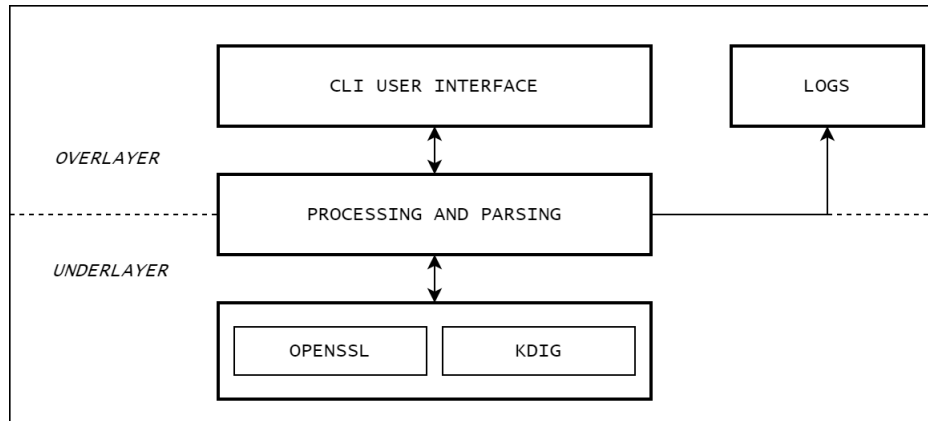
## Chapter 8

# Automated Diagnostic for DNS over Encryption

To make analysis possible on a large scale of data, it was needed to automate a process of diagnostic tools (KDIG and OpenSSL) and order received output into a form suitable for further data processing and statistical analysis. It was expected that the process has to be accelerated by multi-threading to deal with analysis in a reasonable time. Diagnostic tools do not offer automated approach and do not output data into format such as JSON (JavaScript Object Notation) or XML (Extensible Markup Language). Therefore a tool had to be written to process all obtained data with an output of observed properties formatted into line delimited JSON. For this purpose, an automated Diagnostic for DNS over Encryption (D2E) was written in Python to satisfy all mentioned needs.

### 8.1 Overview

D2E in general consist of an underlayer and overlayer. The underlayer presented by KDIG and OpenSSL performs an analysis of selected parameters. The overlayer gives possibility to interact with the D2E through CLI (Command Line Interface). The interaction includes arguments setting, multiple options setting and the progress control of analysis in real-time. In the middle sits a processing part interacting with both layers and exchanging information between them. It launches analysis with KDIG or OpenSSL with different parameters based on the user input and parses received data into a line delimited JSON stored in a file. The processing part makes the diagnostic automate, accelerated and it controls progress of analysis. Fig 8.1 describes components of underlayer and overlayer in the context of D2E tool.



**Figure 8.1:** Architecture of Diagnostic for DNS over Encryption (D2E)

As was already mentioned the DE2 was written in Python 3 and the minimum requirements to launch the tool are shown in Tab. 8.1. Packages from standard Python Library distributed within Python installation are not included.

Software	Version
OpenSSL	1.1.1.1d
KDIG	3.0.5
Python 3	3.7.3
pyOpenSSL	20.0.1
ocsp-checker	1.8.1

**Table 8.1:** Minimum Requirements

The user interface provides several options. The required argument is a file containing list of IPv4 addresses or URLs (one record per line) depending on the diagnostic mode of analysis. The next option `-mode` choose a type of analysis. The *resp* mode extract only legitimate servers providing encryption and evaluate several security and privacy parameters (EDNS0, ECS, DNSSEC and TLS). The second possible argument for mode is *chaos*. This mode tries to identify server with its configured version, hostname or other identifying information about the server that could be misused by attackers. The option `-dnstype` distinguish between DoT and DoH implementation with *tls* or *https* parameters. The last option that user can modify is multi-threading with an option `-threads`. The values can be chosen between 1 to 50 threads depending on the amount of input data and availability of computing resources. The detailed description of every argument can be accessed by help mode `-help`. In Fig. 8.2 the output of a help option describes all possibilities of encrypted DNS analysis with D2E.



```

jan@jan-VirtualBox:~$ python3 d2e.py -h
usage: d2e.py [-h] [-d {tls,https}] [-m {resp,chaos}] [-t THREAD] sourcedata

DNS over TLS (DoT) and DNS over HTTPS (DoH) diagnostic tool

positional arguments:
  sourcedata            Define a path for a list of IPv4 addresses (DoT) or URLs (DoH)
↳ which will be queried.

optional arguments:
  -h, --help            show this help message and exit
  -d {tls,https}, --dnstype {tls,https}
                        Define which encrypted protocol would be used: tls for DNS
↳ over TLS or https for DNS over HTTPS
  -m {resp,chaos}, --mode {resp,chaos}
                        Mode of Operation. Resp option query resolvers in order to get
↳ supported TLS version, cipher suites, certificate and EDNS
                        parameters such as DNSSEC, ECS, EDNS padding. Chaos option
↳ gives an identification through CHAOS class and queries resolvers
                        for version.bind, hostname.bind, authors.bind and id.server
  -t THREAD, --threads THREAD
                        Number of threads used during measurement. Default value is
↳ 10. The minimum value is 1 maximum 50.

```

**Figure 8.2:** User interface with arguments description

## 8.2 Procedure of Analysis

The Fig. 8.3 describes the process of analysis in *resp* mode. In the beginning, data are loaded and based on the number of threads, records are distributed to them. Query control is performed afterwards. If the candidate responds, the evaluation follows the next steps. Otherwise the thread removes the candidate and continues in investigation of the next one in the queue. The received response of successful candidate is captured with full details but evaluated are only parameters regarding TLS, Padding and DNSSEC. Then the next query is sent to get information about ECS handling by recursive resolver. In the last step the certificate from *ServerHello* message is cached and information for further analysis as subject, issuer and validity are extracted. The resulting data from every step are gradually inserted into JSON object and written to the file `resp_<dnstype>-out-phase0.log`. On the other hand, if an error occurs, the thread writes information notice

about a problem into resp\_<dnstype>-err-phase0.log

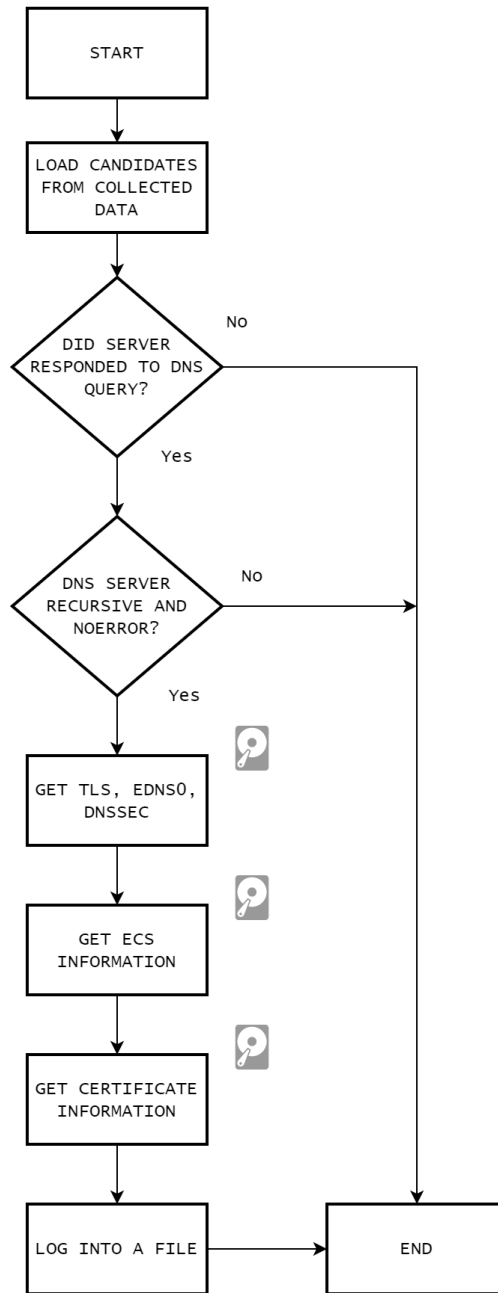
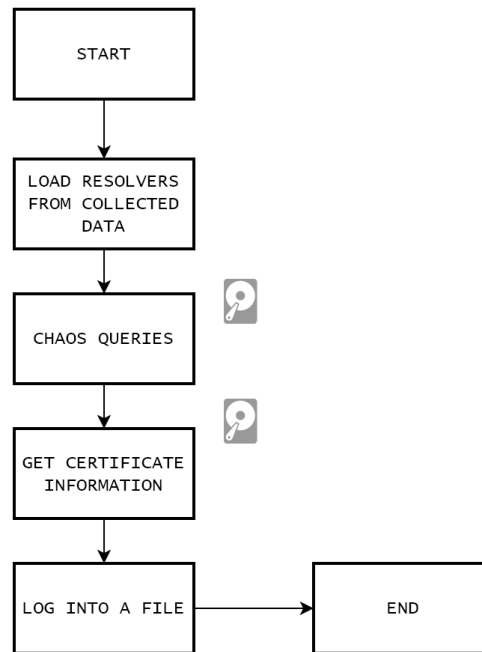


Figure 8.3: Flowchart for resp mode

The *chaos* mode in Fig. 8.4 shows an operation of server identification through DNS. Found legitimate resolvers are input data and targets of queries for CHAOS TXT records hostname.bind, version.bind, authors.bind and id.server. After received response, the certificate information is collected to group multiple queries into a single entity. Data are cached through the whole process of investigation and written into the file chaos\_<dnstype>-out-phase0.log together with error messages which can occur in chaos\_<dnstype>-err-phase0.log.



**Figure 8.4:** Flowchart for chaos mode

An example of an analysis in *resp* mode using DoT and 20 threads to accelerate the process is shown in Fig. 8.5. The output reflects loaded settings and number of candidates. During an analysis, every thread shares information with a user about found resolver and progress to estimate the time of completion.

```
jan@jan-VirtualBox:~$ python3 d2e.py zmap_dot_server.log -m resp -d tls -t 20

[PHASE 0] STARTING DOT RECURSIVE RESOLVER DETECTION      Sun May  9 20:59:56 2021
=====
      a]Choose resursive resolvers only
      b]TLS evaluation: version, cipher suites, certificate
      c]EDNS0 evaluation: ECS, Padding, DNSSEC
=====
Number of candidates: 3384303
Number of threads: 20

20:59:56 Thread 0: Starting
20:59:56 Thread 1: Starting
20:59:56 Thread 2: Starting
20:59:56 Thread 3: Starting
...
...
...
20:59:57 Thread 2: Resolver Found! Number of resolvers: 1; Progress: 0.0006 %
```

**Figure 8.5:** An output of analysis in resp mode

### ■ 8.2.1 Data Output

Data received in the analysis are parsed into line delimited JSON and contain multiple parameters for further evaluation. The Tab. 8.2 describes captured parameters with corresponding data type.

Key	Description	Data Type
IP/URL	server identification based on type of analysis (DoT/DoH)	string
TLS	version, key exchange algorithms, encryption and signatures	string
Status	status of a query	string
Flags	DNS header flags	string
Received bytes	the size of DNS message	string
Time	time of a query	string
Latency	latency in milliseconds (ms)	string
DNSSEC	DNSSEC support	string
EDNS	padding size and EDNS0 information	list
Question	query content including A record to query or TXT record to query	list
Answer	response content with resolution for A record or TXT record	list
OCSP	OCSP URL and information whether certificate is revoked	list
ECS	client subnet information	dictionary
Certificate Subject	subject field of a certificate	dictionary
Certificate Issuer	issuer field of a certificate	dictionary
Expiration	expiration of certificate (control of validity field)	boolean

**Table 8.2:** Fields and description of JSON object

The processed data for 1.1.1.1 captured with D2E are displayed in Fig. 8.6.

```

{
  "IP": "1.1.1.1",
  "TLS": "(TLS1.3)-(ECDHE-X25519)-(ECDSA-SECP256R1-SHA256)-(AES-256-GCM)",
  "Status": "NOERROR;",
  "FLAGS": " Flags: qr rd ra ad; QUERY: 1; ANSWER: 1; AUTHORITY: 0; ADDITIONAL: 1",
  "EDNS": ["Version:", "0;", "flags:", ";", "UDP", "size:", "1232", "B;",
↪ "ext-rcode:", "NOERROR"],
  "Question": ["arnold.dns-diagnostic.cz.", "", "IN", "A"],
  "Answer": ["arnold.dns-diagnostic.cz.", "2186", "IN", "A", "81.95.96.29"],
  "Received bytes": "468",
  "Time": " Time 2021-05-08 21:19:50 CDT",
  "Latency [ms]": "25.0",
  "DNSSEC": "1",
  "ECS": {
    'ecs': 'False',
    'ts': '1620526796.96',
    'recursive': {
      'cc': 'EU',
      'srcip': '141.101.95.30',
      'sport': '13118'
    }
  },
  "Certificate Subject": {
    "C": "US",
    "ST": "California",
    "L": "San Francisco",
    "O": "Cloudflare, Inc.",
    "CN": "cloudflare-dns.com"
  },
  "Certificate issuer": {
    "C": "US",
    "O": "DigiCert Inc",
    "CN": "DigiCert TLS Hybrid ECC SHA384 2020 CA1"
  },
  "Expiration": false,
  "OCSP": ["Host: cloudflare-dns.com:None", "OCSP URL: http://ocsp.digicert.com",
↪ "OCSP Status: GOOD"]}

```

**Figure 8.6:** An example of 1.1.1.1 JSON record

## Chapter 9

### Results of Measurement

The following chapter describes discovered servers supporting encrypted DNS. Further processing and evaluation of multiple parameters regarding DNS and TLS protocol were performed with a diagnostic tool. The observed parameters are further discussed.

#### 9.1 Collected Data

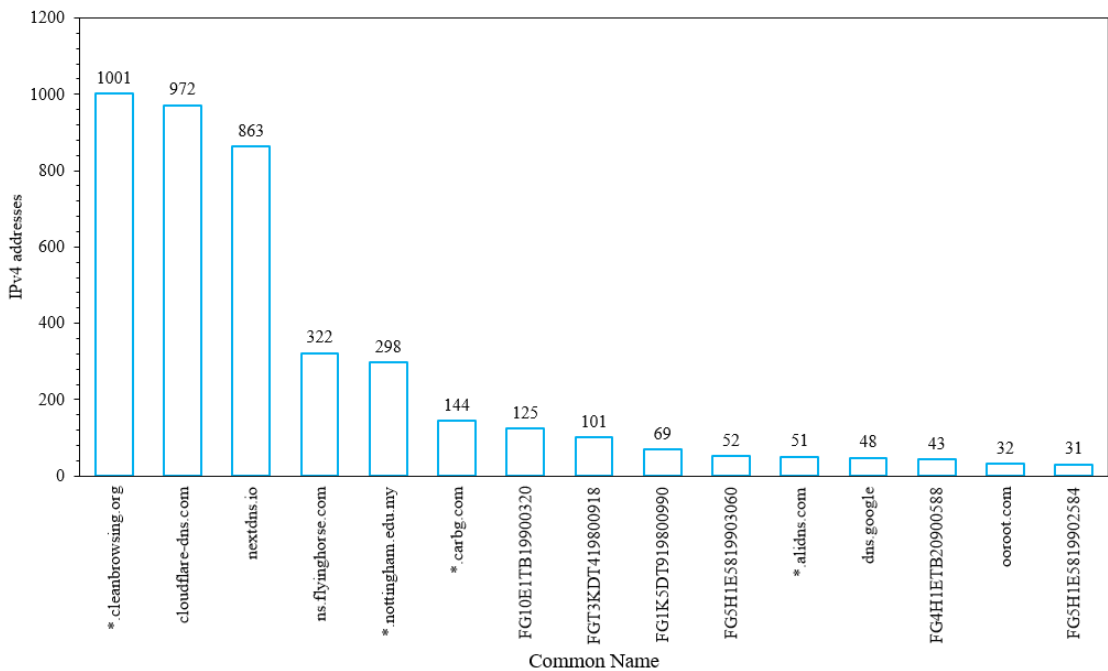
Data collection of DoT servers was carried out from 30th of April to 2nd of May 2021 from the Czech Technical University, AS2852. In total, 3 384 303 number of IPv4 addresses exposed port 853. The filtration process took place afterwards. Only those servers which indicate status NOERROR and provide resolution to the correct A record were considered further. Otherwise, some parameters would not be available for evaluation. Tab. 9.1 represents received status from responding servers for DoT query. Thus total number of 10 048 IPv4 supporting DoT and providing correct resolution was discovered. The majority of SERVERFAIL, FORMERR, NXDOMAIN or NOTZONE statuses could be caused by misconfiguration of resolvers. Relatively high percent of REFUSED statuses may be caused by geolocation restricted policies (e.g blocking requests to US location from Europe), exceeding rate limits of queries or denied access to servers that offer paid services such as DNS security protection or parental control. It is also worth mentioning that discovered number does not reflect true amount of physical instances running DoT, because of anycast routing which could possibly hide many instances behind a single address.

NOERROR*	FORMERR	SERVERFAIL	NXDOMAIN	REFUSED	NOTZONE
10 048	68	745	671	946	4
76.486 %	0.517 %	5.671 %	5.107 %	7.201 %	0.030 %

\*resolution successful

**Table 9.1:** Discovered IPv4 addresses

To unify individual providers, the CN in Certificate Subject field was used to identify the representation as shown in Fig. 9.1. The leading provider has become *cleanbrowsing.org*, *cloudflare-dns.com* and *nextdns.io*. The results also showed relatively large quantity of CN starting with 'FG'. Those certificates belong to FortiGate NGFW (Next-Generation Firewall) devices most likely defending and proxying DNS queries to internal DNS servers. Those servers are not directly connected with one entity but rather belonging to multiple global providers or ISPs, because addresses belong to different public ranges registered by different organizations. If those certificates with Fortigate CN issued by Fortinet CA should be grouped, they represent 32 % of total discovered DoT servers.



**Figure 9.1:** The most extended providers of DoT grouped by CN

Data collected for DoH servers depended on online source providing list of URLs [59] and previous data collection, because support of both encrypted DNS protocols was expected by some providers. The input variable for DoH analysis is URL. As it can be seen in Tab. 9.2 together from an online source and the previous Internet-wide



survey was collected 547 DoH servers with correct resolution and NOERROR status. The number of failed resolution was under 1 % and 3.156 % of resolvers refused to reply. Compare to DoT, the failed attempts are much lower. It could be linked with the fact that unsuccessful resolution results in service disruption which is not the case with DoT using Opportunistic Privacy Profile offering backup cleartext resolution.

NOERROR*	REFUSED	SERVFAIL
547	19	5
90.864 %	3.156 %	0.831 %

\*resolution successful

**Table 9.2:** Discovered URLs

The CN group of providers was not made, because URL based approach does not provide accurate distribution of providers as IPv4 addresses did. If URL from specific provider occurred more than once, it was offering different type of services such as standard mode and mode blocking potential malware and ads.

## 9.2 DNS Layer

The following section describes the analysed parameters of encrypted DNS protocol. At first, the length of message from responding implementations of DoT and DoH was taken. Based on EDNS0 option and message size, it was decided whether padding or additional information are carried in DNS messages. Fig. 9.2 and Fig 9.3 show discovered message sizes for DoT and DoH. It can be seen that in most cases the message length is 69 bytes which does not include padding. In DoH protocol, significant number of messages has length of 93 and 104 bytes. These messages held ECS information in a response. Only 23.537 % of DoT servers provided padding. Even lower percentage of padded responses (9.872 %) was captured with DoH protocol. The size and distribution of all padded messages is depicted in Tab. 9.3

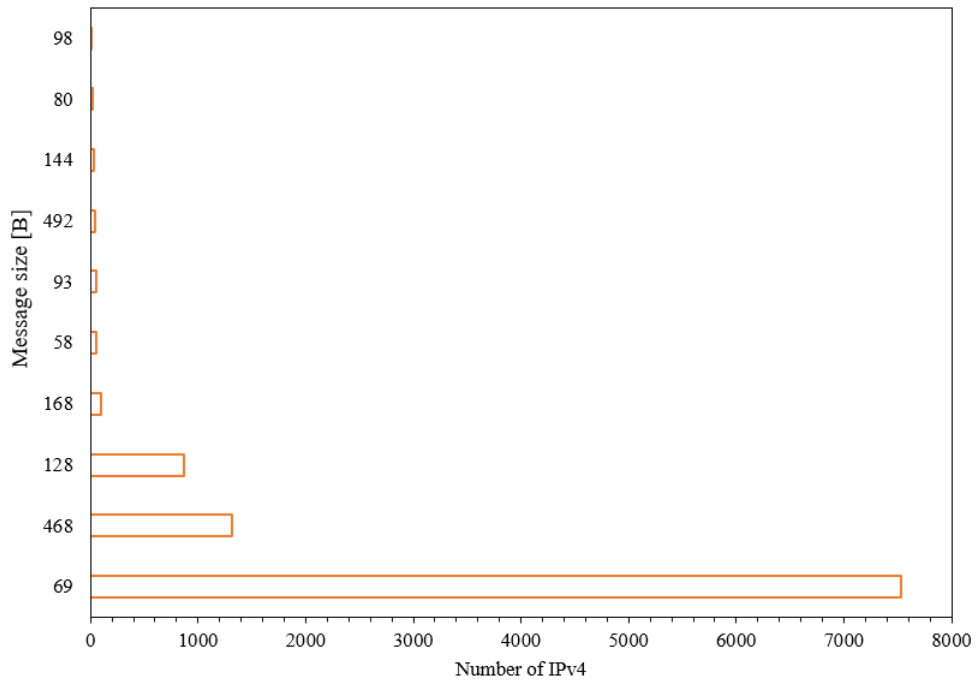


Figure 9.2: Length of DoT messages

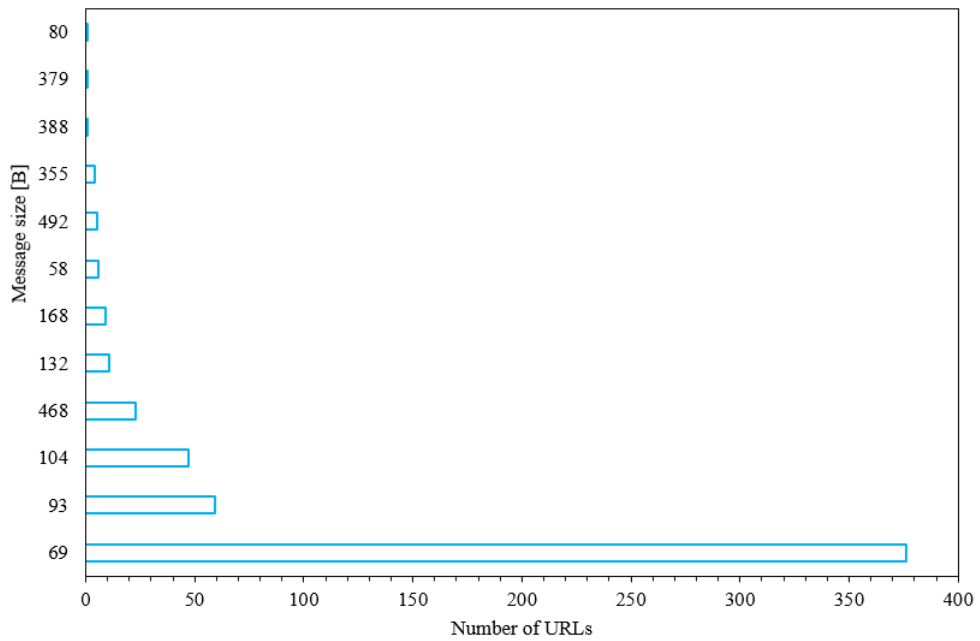


Figure 9.3: Length of DoH messages

In Tab. 9.3 can be also seen that recommended padding size of 468 bytes is the most popular among padded messages in DoT resolvers. The providers from top 15 introduced

in Fig. 9.1 did not respect recommended padding completely, only *cloudflare-dns.com*, *dns.google* and *ooroot.com* used messages padded to 468 bytes. The second most popular padding of 128 bytes implemented servers from *nextdns.io*. Nevertheless, it is padding recommended for clients based on RFC.

DoH implementation of padding is very rare, still size of 468 bytes is the most preferable. The following providers are compliant with padding recommendation in DoH: *carolinareaper.heeremans.net*, *cloudflare-dns.com*, *dns.emeraldionion.org*, *dns.google*, *dns.lvfrfn.in.ua*, *adguard.ekhozie.dynu.net*, *green.runapp.bid*, *hermes.ohai.ca*, *odvr.nic.cz*, *doh.statpro.com*, *stream2.radiocfm.ro*, *dns.thalheim.io* *doh.dev.andronkyr.com*, *dns.sindominio.net*, *youbak.com*, *adguard.brais.dev*, *basic.bravedns.com*.

Message size with padding	DoT	DoH
492 B	0.448 %	0.914 %
481 B	0.010 %	-
468 B	13.037 %	4.205 %
388 B	-	0.183 %
355 B	0.080 %	0.731 %
379 B	0.050 %	0.183 %
168 B	0.965 %	1.645 %
144 B	0.328 %	-
132 B	-	2.011 %
128 B	8.691 %	-

**Table 9.3:** Messages containing padding

As the length of message is increased with additional padding, the latency of exchanged messages was compared. The comparison was performed to find if additional message size influences the time of delivery. It is important to mention that the measured latency covered an initial message exchange which includes an additional cost of a channel establishment. As Fig. 9.4 and Fig. 9.5 present, the latency of padded messages surprisingly showed very good results even better than not padded messages. The padded messages with 468 bytes in DoT protocol achieved mean value of 152.352 ms. DoH protocol reaches the mean value of 208.456 ms with 468 byte messages. The results could be influenced by the fact that the padding is supported by few providers which could provide fast cryptography algorithms or close geolocation with anycast routing. Thus, they can afford padding. It could be also possible that the increased message size of an application layer can be insignificant to the latency caused by under layer channel

establishment mechanisms. It can be concluded that from the location the measurement was made, the padding of initial message exchange does not influence latency. However to measure whether padding could cause increased latency and worsened user experience, the measurement of multiple queries through established channel had to be made and compared to not padded messages.

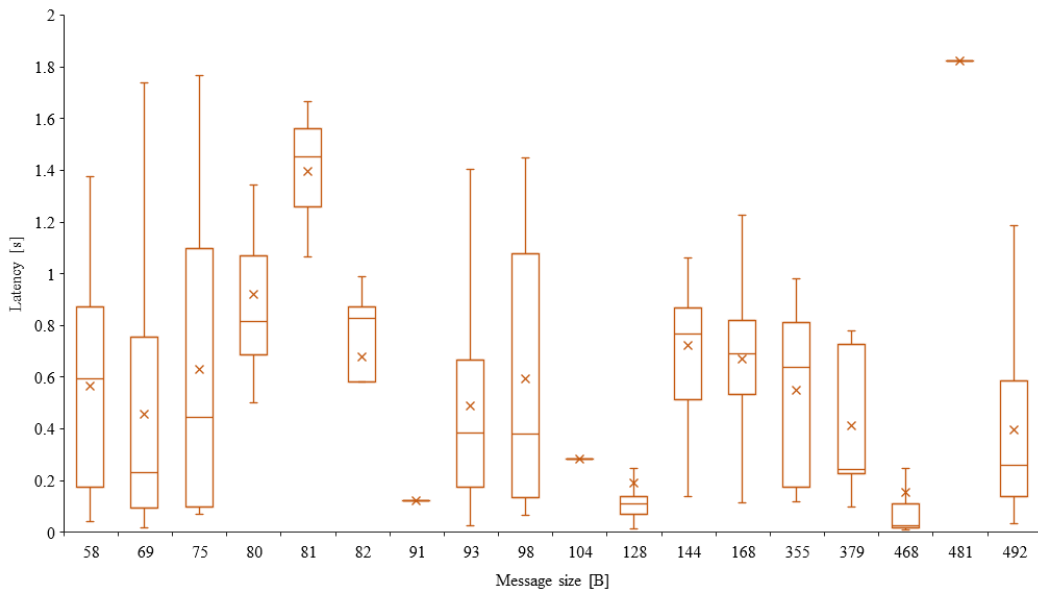


Figure 9.4: Latency of different DoT message size

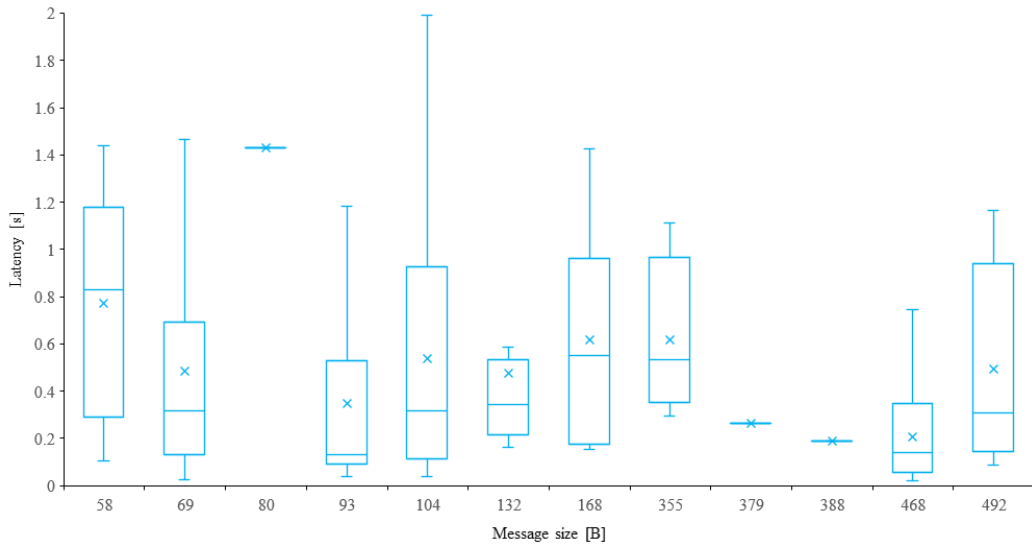


Figure 9.5: Latency of different DoH message size

Since the messages continue to authoritative servers it is important to ensure that the complete path is secured as much as possible. For this reason, DNSSEC is an integral part of secure message exchange but sometimes omitted with regards to encrypted DNS. The Tab. 9.4 presents support amongst encrypted resolvers. Both versions provided similar results with more than 10 % of resolvers which did not support DNSSEC.

	DNSSEC supported	DNSSEC not supported
DoT	84.942 %	15.058 %
DoH	87.021 %	12.979 %

**Table 9.4:** DNSSEC support in encrypted DNS

The ECS support was evaluated next. It can be seen in Tab. 9.5 that was discovered 16.262 % of DoT servers supporting ECS extensions but only 1.234 % revealed the source subnet with network mask /24. Some of the responses including those from servers belonging to *.alidns.com* carried content of ECS with address located in EU. Most likely they did it to speed up the process of resolution but not revealing client information. Minority of largest providers such as *.quad9.net* and *dns.google* with 8.8.8.8 and 8.8.4.4 address propagated information about the source subnet of a query to following authoritative servers. The majority of servers supporting ECS inserted in an extension its own subnet which is desirable for the client privacy and it can improve performance of response delivery. The similar situation is with DoH, where ECS is more supported and public address revealed 6.764 % providers. It is also worth mentioning that ECS was in many cases applied by smaller providers with poor geolocation coverage.

	ECS support	ECS support with revealed address
DoT	16.262 %	1.234 %
DoH	19.744 %	6.764 %

**Table 9.5:** ECS support in encrypted DNS

Since encrypted DNS inherited the same structure of traditional DNS, it was possible to try server identification by querying TXT records of CHAOS class. The queried TXT records were following: *authors.bind*, *hostname.bind*, *version.bind*, *id.server*. To reflect as much data as possible, the queried servers also included those which did not provide correct resolution and returned different status than NOERROR. Firstly, the *authors.bind* was queried and did not reveal any specific information for both DoT and

DoH. Only default values and text offering paid subscription of services were found.

For `id.server` query responded 41.929 % of DoT servers and 29.457 % of DoH servers. In most cases, the information gathered included specification of server location or hostnames. This information was very similar to those collected from `hostname.bind` query which contains almost identical type of information not considering as sensitive.

The last and the most important part was revealed with `version.bind` query. 41.860 % of DoT resolvers and 38.915 % of DoH resolvers responded. The server's responses include combination of custom messages modified by administrator and default values identifying software and its version. 10.830 % of DoT resolvers provide detailed identification of server including software version and in some cases an OS (Operating System) on which server is launched. DoH servers revealed its identification in 21.550 % cases including the same information as described with DoT. This information can be possibly misused by cybercriminals if vulnerabilities of systems are found or specifically targeted attack is prepared. Therefore identities of providers and their specific software versions of encrypted DNS servers are not disclosed, only distribution of software implementations are presented in Fig 9.6 and Fig. 9.7.

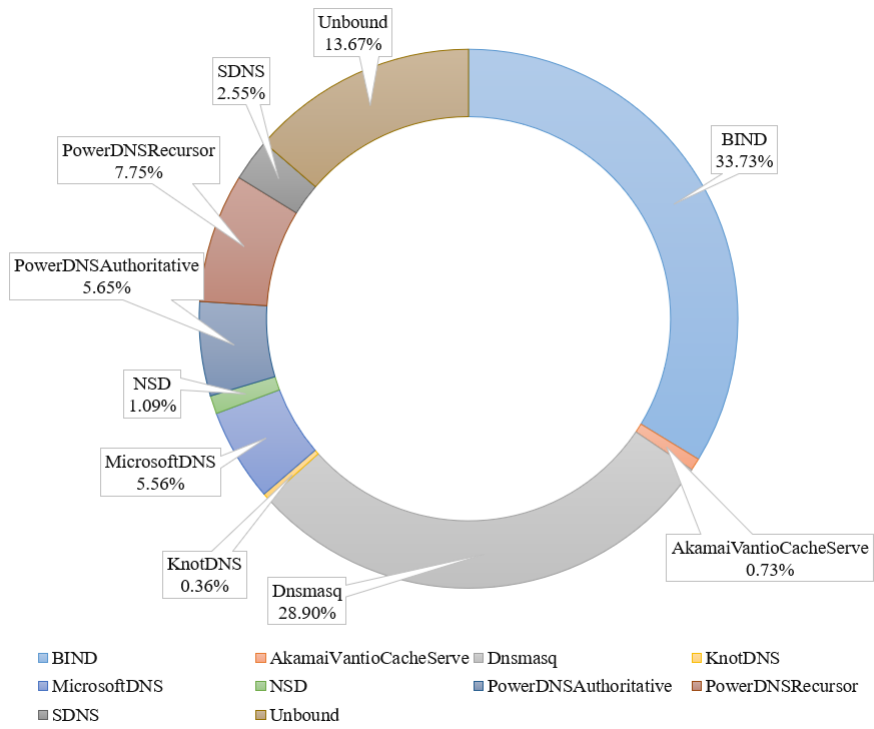


Figure 9.6: Software implementation of discovered DoT servers

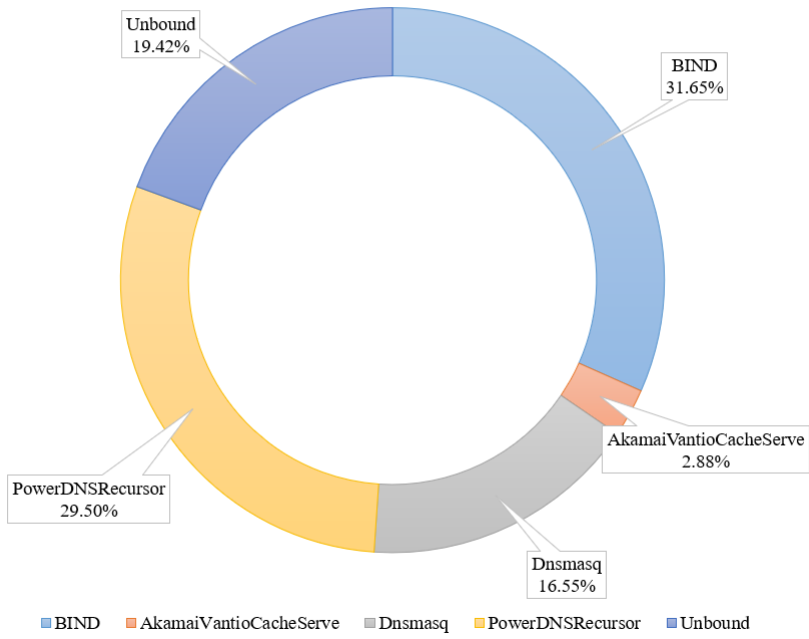


Figure 9.7: Software implementation of discovered DoH servers

### 9.3 TLS Layer

As a security of exchanged messages attaches much importance to TLS. Several documents and recommendation from NIST (National Institute of Standards and Technology) and NUKIB (National Cyber and Information Security Agency) are therefore followed in performed analysis [65] [66].

Beginning with the version of the protocol, a TLS evaluation of DoT resolvers showed that in most cases negotiated version with a server was TLS 1.3. The Tab 9.6 displays complete results an the portion of TLS 1.2. In total, only 0.139 % of servers responded with deprecated TLS 1.0. DoH negotiated even higher number of TLS 1.3 sessions and no deprecated versions were found.

	TLS 1.0	TLS 1.2	TLS 1.3
DoT	14 0.139 %	1615 16.073 %	8419 83.788 %
DoH	- -	71 12.980 %	476 87.020 %

**Table 9.6:** TLS version across encrypted DNS

Encrypted DNS depends on PKI and established a trust to server needs to be verified with several options such as validity, revocation of certificates with OCSP or CRL. Therefore, it was measured whether certificates are not expired and support OCSP which should be preferred over CRL according to NIST [66]. The results from analysis showed that 7.205 % of certificates captured in *Certificate* message were expired and only 32.036 % were successfully verified with OCSP. The certificate validity and revocation of DoH servers provided better results than with DoT. 4.734 % of certificates were found to be expired but 75.868 % were successfully verified with OCSP. The result showed that quite a high number of providers did not provide satisfying prove of trust.

The following Tab. 9.7 and Tab. 9.8 present a summary of the most negotiated algorithms with DoT and DoH servers during the analysis. TLS 1.3 also provided detailed information about the signature algorithms and elliptic curves, whereas TLS 1.2 is limited to algorithm mode only. The weak and not recommended ciphers were found in DoT protocol very rarely such as AES-256-CBC (0.010 %), AES-128-CBC (0.149 %), SHA-1 (0.159 %) or 2048-bit DHE key exchange (0.020 %). DoH did not suffer from inappropriate and weak ciphers. It is still possible that the providers of encrypted DNS could implement older algorithms for broad availability of older devices, but if the



client support secure algorithms as was the case of measurement, most of the servers will provide security with recommended ciphers.

Parameters	Distribution
(TLS1.3)-(ECDHE-SECP256R1)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)	50.975%
(TLS1.3)-(ECDHE-X25519)-(ECDSA-SECP256R1-SHA256)-(AES-256-GCM)	18.780%
(TLS1.2)-(ECDHE-SECP256R1)-(RSA-SHA256)-(AES-256-GCM)	14.859%
(TLS1.3)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)	8.877%
(TLS1.3)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(AES-128-GCM)	1.194%
(TLS1.3)-(ECDHE-SECP256R1)-(ECDSA-SECP384R1-SHA384)-(AES-256-GCM)	1.115 %
(TLS1.3)-(ECDHE-SECP256R1)-(ECDSA-SECP256R1-SHA256)-(AES-256-GCM)	1.005 %
(TLS1.2)-(ECDHE-SECP256R1)-(RSA-SHA512)-(AES-256-GCM)	0.478 %
(TLS1.3)-(ECDHE-SECP256R1)-(RSA-PSS-RSAE-SHA256)-(AES-128-GCM)	0.438 %
(TLS1.3)-(ECDHE-X25519)-(ECDSA-SECP384R1-SHA384)-(AES-256-GCM)	0.358 %

**Table 9.7:** Negotiated algorithms with DoT servers

Parameters	Distribution
(TLS1.3)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(AES-128-GCM)	29.982 %
(TLS1.3)-(ECDHE-SECP256R1)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)	25.594%
(TLS1.3)-(ECDHE-SECP256R1)-(ECDSA-SECP384R1-SHA384)-(AES-256-GCM)	5.850%
(TLS1.3)-(ECDHE-X25519)-(ECDSA-SECP256R1-SHA256)-(AES-256-GCM)	5.667%
(TLS1.2)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(AES-128-GCM)	4.570%
(TLS1.3)-(ECDHE-X25519)-(ECDSA-SECP256R1-SHA256)-(AES-128-GCM)	4.388%
(TLS1.3)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(CHACHA20-POLY1305)	4.205%
(TLS1.3)-(ECDHE-SECP384R1)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)	3.108%
(TLS1.2)-(ECDHE-SECP256R1)-(RSA-SHA256)-(AES-256-GCM)	2.194 %
(TLS1.3)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)	2.194 %

**Table 9.8:** Negotiated algorithms with DoH servers

The next section focuses on each algorithm negotiated with server. The key establishment in Tab. 9.9 is the first part. In total, the ECDHE key exchange is unsurprisingly the most used method and only the elliptic curves chosen by server are diverse. The fact is that the algorithm is mandatory for TLS 1.3 a recommended to implement because of security and performance enhancements. The implementation of elliptic curves also differs between encrypted DNS. The DoT servers mostly implement SECP256R1 also known as NIST P-256 with 256-bit prime providing sufficient security and belonging to NSA Suite B of cryptography. Instead DoH servers preferred more likely X25519 also

known as Daniel Bernstein’s Curve25519 curve which is an alternative to NIST P-256. It is considered to be more trustworthy for some security administrators than the NIST curves because of its transparent justification and institutional independence. Increased prime of key exchange methods which gives even better security properties is occasional in both cases, most likely because of introduced delay in channel establishment.

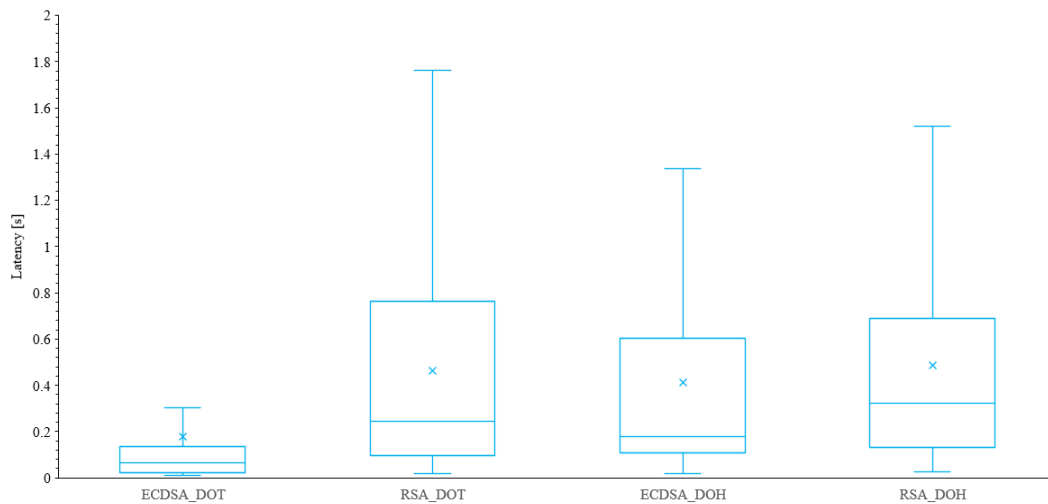
Protocol	Key Exchange	Distribution
DoT	ECDHE-SECP256R1	69.616%
	ECDHE-X25519	30.135%
	ECDHE-SECP384R1	0.159%
	ECDHE-SECP521R1	0.060%
	DHE-CUSTOM2048	0.020%
DoH	ECDHE-X25519	54.662%
	ECDHE-SECP256R1	40.037%
	ECDHE-SECP384R1	3.839%
	ECDHE-SECP521R1	1.463%

**Table 9.9:** Key exchange algorithms

The authenticity of exchanged messages is a necessary part of handshake process and it was examined below. Digital signature algorithms provided authentication in most cases by RSA-PSS (RSA with Probabilistic Signature Scheme). It is recommended and more robust algorithm than PKCS1v1.5 which was found to be vulnerable to Bleichenbacher oracle attacks. ECDSA (Elliptic Curve Digital Signature Algorithm) is the second most used signature. It provides much shorter key lengths with the same level of security and therefore it can be a good choice for latency-critical applications as encrypted DNS. As an example, the RSA key lengths of 2048 bits correspond with the security of 256 bits key lengths of ECDSA. The previous work showed that a key generation time for ECDSA is significantly faster than RSA, but the verification time took shorter time with RSA due to complex operations that ECDSA implements [67]. The recommended curves with ECDSA by NIST are followed (SECP384R1 and SECP256R1) with discovered implementations. The Tab. 9.10 shows majority of providers implementing RSA signatures in both encrypted protocols. It is also interesting that in both DoT and DoH, *cloudflare-dns.com* achieved the best result in latency with ECDSA. The presumption of dependency of signatures and latency of DNS message also confirmed Fig. 9.8 showing shorter time achieved in DoT and DoH with ECDSA. Therefore, an influence of signature algorithms and latency in encrypted DNS could be suggestion for further more detailed studies.

Protocol	Digital Signature	Distribution
DoT	ERSA-PSS-RSAE-SHA256	61.883 %
	ECDSA-SECP256R1-SHA256	20.422 %
	RSA-SHA256	15.008 %
	ECDSA-SECP384R1-SHA384	1.582 %
	RSA-SHA512	0.697 %
	ECDSA-SHA256	0.189 %
	ECDSA-SHA512	0.040 %
	RSA-PSS-RSAE-SHA512	0.020 %
ECDSA-SECP521R1-SHA512	0.010 %	
DoH	RSA-PSS-RSAE-SHA256	72.445 %
	ECDSA-SECP256R1-SHA256	12.226 %
	ECDSA-SECP384R1-SHA384	8.212 %
	RSA-SHA512	3.467 %
	RSA-SHA256	2.190 %
	ECDSA-SHA512	0.730 %
	ECDSA-SHA256	0.547 %

**Table 9.10:** Digital Signature Algorithms



**Figure 9.8:** Digital signature latency

The last part consists of data encryption. From the Tab. 9.11 it can be seen that DoT servers strongly ensures security of exchanges messages with the GCM (Galois Counter Mode) and 256 bit key length. GCM is an AEAD algorithm that is used for confidentiality, integrity, message authentication and it is recommended by NIST. The symmetric encryption in DoH is also based on GCM. Nevertheless, negotiated keys with

256 bits did not cover as much proportion as with DoT. It is important to mention that other AES modes such as CBC (Cipher Block Chaining) mode, vulnerable to plain-text attacks [68] covered less than 1 % of discovered servers. Trailing hash indicates the hashing algorithm that is used with the HKDF (HMAC Key Derivation Function) during key derivation in TLS 1.3 and PRF in older version. Only the same servers which offered weak symmetric encryption did negotiated weak hash algorithms with SHA-1 which is considered to be insecure [69].

Protocol	Encryption and Hash Algorithms	Distribution
DoT	AES-256-GCM-SHA384	97.193 %
	AES-128-GCM-SHA256	2.040 %
	CHACHA20-POLY1305	0.607 %
	AES-128-CBC-SHA1	0.149 %
	AES-256-CBC-SHA1	0.010 %
DoH	AES-256-GCM	51.277 %
	AES-128-GCM	41.788 %
	CHACHA20-POLY1305	6.752 %

**Table 9.11:** Encryption and hash algorithms



## Chapter 10

### Conclusion

The Master's Thesis described DNS, extensions mechanisms and parameters which are still present in DoT and DoH. These extensions included ECS or padding. Increased security defending against on-path observer was shown and importance of these counter-measures was compared with different situations across the world. It was concluded that the impact of encrypted DNS differs and the major concerns that encryption prevents were found to be user data monetization, information censorship and DNS hijacking.

It was also necessary to draw attention to a new design of encrypted DNS protocols. Because of its stateful characteristic and HTTP/2, the client identification by the service providers would be possible. DNS request can reveal rich information about the user activity, hence if these data are not available to observer anymore, it is still important for operators to keep this information safe, not misusing it for business purposes and prevent data loss. For that reason, the thesis put attention to the security compliance of service providers who operate encrypted DNS and a survey of public open resolvers was performed.

From a technical point of view, DNS and TLS protocol included in encrypted DNS were analysed with an automated D2E tool written for large scale data measurement. It was discovered that some of the leading providers protect message against correlation analysis with messages padded to 468 bytes. In general, the recommendation is not followed and only 23.537 % of DoT and 9.872 % of DoH resolvers provided padding option with variable size. The DNSSEC support between resolver and authoritative name server was also measured to provide the maximum possible protection. The result showed that still more than 10 % of DoT and DoH providers did not authenticate the responses, thus not protect the whole message path. ECS was the next parameter of interest that could leak client subnet information to authoritative servers. Its support was found in 1.234 % cases in DoT and 6.764 % in DoH. It was also discovered that some servers provided summarized address in ECS to get a more geographically favourable

response and not revealing client subnet information. In the last part of DNS exploration was discovered that reconnaissance attacks through encrypted DNS are possible and can reveal software and version of running resolver. This finding could cause harm if vulnerabilities in software occur. It is also very likely that some providers did not implement true DoT or DoH servers. Several found software versions did not support encrypted DNS and with a quite significant portion (32 %) of certificates captured in DoT, the firewall was found in a path. These implementations may terminate traditional DNS in a local network on firewall device and provide encryption to the public clients only. This solution should be functional for DoT but could have performance issues with DoH which use the value 0 for ID of transaction in order to provide maximize HTTP cache friendliness.

TLS protocol encapsulating both versions of encrypted DNS showed satisfying results with support of TLS 1.3 in 83 % of DoT and 87 % of DoH. Negotiated parameters were following recommendations and weak algorithms were exceptions. The variability of elliptic curves in key exchange algorithms was discussed and performance impact of digital signatures showed shorter time of message exchange with ECDSA. The weak part of TLS in DoT and DoH was related to PKI and certificates which provided poor score of reliability with expired dates and low number of OCSP support. It was also discovered that DoH which does not provide fall back mechanism and resolution fails if the channel establishment is not possible (expired, revoked or untrusted certificates) achieved better OCSP support (75.868 %) and fewer expired certificates (4.734 %).

In a conclusion, it is suggested for service providers to follow security recommendations published by RFC, NIST and other security professionals and defend their systems with sufficient algorithms. The weakest discovered parts of protection included insufficient implementation of padding, possible server fingerprinting and weak implementation of PKI. Except technical specifications and recommendation, it would be also relevant to cooperate with lawyers and establish appropriate regulations protecting user data passing through resolvers with legislation.



## Bibliography

1. LIU, Cricket; ALBITZ, Paul. *DNS and BIND*. Fifth edition. Sebastopol, CA: O'Reilly Media, 2006. ISBN 978-0-596-10057-5.
2. *How DNS Works* / *ITGeared.com* [online]. 2012 [visited on 2021-04-30]. Available from: <https://www.itgeared.com/articles/1354-domain-name-system-dns-tutorial-overview/>.
3. *Introduction to DNS Privacy* [Internet Society] [online] [visited on 2021-04-30]. Available from: <https://www.internetsociety.org/resources/deplo360/dns-privacy/intro/>.
4. *What Is DNS? / How DNS Works* [Cloudflare] [online] [visited on 2021-04-30]. Available from: <https://www.cloudflare.com/learning/dns/what-is-dns/>.
5. *What is DNS? – Introduction to DNS - AWS* [Amazon Web Services, Inc.] [online] [visited on 2021-04-30]. Available from: <https://aws.amazon.com/route53/what-is-dns/>.
6. RIKITAKE, Kenji; NAKAO, Koji; NOGAWA, Hiroki; SHIMOJO, Shinji. *T/TCP for DNS: A Performance and Security Analysis*. 2021.
7. *Chapter 11. Name Resolution and the Domain Name System (DNS) - Shichao's Notes* [online] [visited on 2021-04-30]. Available from: <https://notes.shichao.io/tcpv1/ch11/#resource-record-types>.
8. MOCKAPETRIS, P. V. *Domain names - implementation and specification* [online] [visited on 2021-04-30]. Available from: <https://tools.ietf.org/html/rfc1035>.
9. DAMAS, Joao; GRAFF, Michael; VIXIE, Paul. *Extension Mechanisms for DNS (EDNS(0))* [online] [visited on 2021-04-30]. Available from: <https://tools.ietf.org/html/rfc6891>.

10. MAYRHOFER, Alexander. *The EDNS(0) Padding Option* [online] [visited on 2021-04-30]. Available from: <https://tools.ietf.org/html/rfc7830>.
11. KINTIS, Panagiotis; NADJI, Yacin; DAGON, David; FARRELL, Michael; ANTONAKAKIS, Manos. Understanding the Privacy Implications of ECS. 2016. Available also from: [https://astrolavos.gatech.edu/articles/dimva16\\_ecs.pdf](https://astrolavos.gatech.edu/articles/dimva16_ecs.pdf).
12. *Rozhovor: Meziplanetární internet je reálný projekt, říká Vinton Cerf - iDNES.cz* [online] [visited on 2021-04-30]. Available from: [https://www.idnes.cz/technet/internet/rozhovor-meziplanetarni-internet-je-realny-projekt-rika-vinton-cerf.A070406\\_201357\\_sw\\_internet\\_pka](https://www.idnes.cz/technet/internet/rozhovor-meziplanetarni-internet-je-realny-projekt-rika-vinton-cerf.A070406_201357_sw_internet_pka).
13. LISKA, Allan; STOWE, Geoffrey. *DNS Security: Defending the Domain Name System*. 1st edition. Amsterdam ; Boston: Syngress, 2016. ISBN 978-0-12-803306-7.
14. *DNS Amplification Attacks / CISA* [online] [visited on 2021-04-30]. Available from: <https://us-cert.cisa.gov/ncas/alerts/TA13-088A>.
15. *What is Anycast DNS? / How Anycast Works With DNS* [Cloudflare] [online] [visited on 2021-04-30]. Available from: <https://www.cloudflare.com/learning/dns/what-is-anycast-dns/>.
16. *DNSSEC – What Is It and Why Is It Important? - ICANN* [online] [visited on 2021-04-30]. Available from: <https://www.icann.org/resources/pages/dnssec-what-is-it-why-important-2019-03-05-en>.
17. WELLINGTON, Brian. *Redefinition of DNS Authenticated Data (AD) bit* [online] [visited on 2021-04-30]. Available from: <https://tools.ietf.org/html/rfc3655>.
18. *SAD DNS Explained* [The Cloudflare Blog] [online]. 2020-11-13 [visited on 2021-04-30]. Available from: <https://blog.cloudflare.com/sad-dns-explained/>.
19. *ISP selling data: why you should actually care* [CyberNews] [online]. 2019-10-03 [visited on 2021-04-30]. Available from: <https://cybernews.com/privacy/isp-selling-data-why-you-should-actually-care/>.
20. WEAVER, Nicholas; KREIBICH, Christian; PAXSON, Vern. Redirecting DNS for Ads and Profit. 2011, p. 6. Available also from: <https://www.icsi.berkeley.edu/pubs/networking/redirectingdnsforads11.pdf>.
21. *Content filtering by UK ISPs* [online] [visited on 2021-04-30]. Available from: [https://wiki.openrightsgroup.org/wiki/Content\\_filtering\\_by\\_UK\\_ISPs](https://wiki.openrightsgroup.org/wiki/Content_filtering_by_UK_ISPs).



22. *Deconstructing the Great Firewall of China* [online] [visited on 2021-04-30]. Available from: <https://www.thousandeyes.com/blog/deconstructing-great-firewall-china/>.
23. *The Fight Over Encrypted DNS: Explained - IEEE Spectrum* [IEEE Spectrum: Technology, Engineering, and Science News] [online] [visited on 2021-04-30]. Available from: <https://spectrum.ieee.org/tech-talk/telecom/security/the-fight-over-encrypted-dns-boils-over>.
24. *Cloud Enterprise Network Security | Cisco Umbrella* [online] [visited on 2021-04-30]. Available from: <https://umbrella.cisco.com/>.
25. LYNCH, Vincent. *Deprecating TLS 1.0 & 1.1 | DigiCert Blog* [DigiCert] [online]. 2018-03-21 [visited on 2021-05-01]. Available from: </blog/deprecating-tls-1-0-and-1-1/>. Section: Announcements.
26. *Qualys SSL Labs - SSL Pulse* [online] [visited on 2021-05-01]. Available from: <https://www.ssllabs.com/ssl-pulse/>.
27. RISTIC, Ivan. *Bulletproof SSL and TLS: Understanding and Deploying SSL/TLS and PKI to Secure Servers and Web Applications*. London: Feisty Duck, 2014. ISBN 978-1-907117-04-6.
28. BURDA, Karel. *Kryptografie okolo nás*. 2019, p. 132. ISBN 978-80-88168-49-2.
29. ALASHWALI, Eman Salem; RASMUSSEN, Kasper. What's in a Downgrade? A Taxonomy of Downgrade Attacks in the TLS Protocol and Application Protocols Using TLS. *arXiv:1809.05681 [cs]* [online]. 2019 [visited on 2021-05-01]. Available from arXiv: 1809.05681.
30. *DHE* [online] [visited on 2021-05-01]. Available from: <https://asecuritysite.com/encryption/dhe>.
31. *Certificate Transparency : Certificate Transparency* [online] [visited on 2021-05-18]. Available from: <https://certificate.transparency.dev/>.
32. *openssl/openssl* [online]. OpenSSL, 2021 [visited on 2021-05-01]. Available from: <https://github.com/openssl/openssl>. original-date: 2013-01-15T22:34:48Z.
33. *A Detailed Look at RFC 8446 (a.k.a. TLS 1.3)* [The Cloudflare Blog] [online]. 2018-08-10 [visited on 2021-05-01]. Available from: <https://blog.cloudflare.com/rfc-8446-aka-tls-1-3/>.
34. *DNSCrypt version 2 - Official Project Home Page* [online] [visited on 2021-04-30]. Available from: <https://dnscrypt.info/>.



46. *DNSFilter: DoH Isn't Better, It's What Google Likes: DNS-over-TLS* [online] [visited on 2021-05-02]. Available from: <https://www.dnsfilter.com/blog/dns-over-tls>.
47. BÖTTGER, Timm; CUADRADO, Felix; ANTICHI, Gianni; FERNANDES, Eder Leão; TYSON, Gareth; CASTRO, Ignacio; UHLIG, Steve. An Empirical Study of the Cost of DNS-over-HTTPS. In: *Proceedings of the Internet Measurement Conference* [online]. Amsterdam Netherlands: ACM, 2019, pp. 15–21 [visited on 2021-05-02]. ISBN 978-1-4503-6948-0. Available from DOI: 10.1145/3355369.3355575.
48. SIBY, Sandra; JUAREZ, Marc; DIAZ, Claudia; VALLINA-RODRIGUEZ, Narseo; TRONCOSO, Carmela. Encrypted DNS = Privacy? A Traffic Analysis Perspective. 2019, p. 18.
49. *SensePost / Waiting for godoh* [online] [visited on 2021-04-30]. Available from: <https://sensepost.com/blog/2018/waiting-for-godoh/>.
50. CIMPANU, Catalin. *Iranian hacker group becomes first known APT to weaponize DNS-over-HTTPS (DoH)* [ZDNet] [online] [visited on 2021-04-30]. Available from: <https://www.zdnet.com/article/iranian-hacker-group-becomes-first-known-apt-to-weaponize-dns-over-https-doh/>.
51. VEKSHIN, Dmitrii; HYNEK, Karel; CEJKA, Tomas. DoH Insight: detecting DNS over HTTPS by machine learning. In: *Proceedings of the 15th International Conference on Availability, Reliability and Security* [online]. Virtual Event Ireland: ACM, 2020, pp. 1–8 [visited on 2021-05-02]. ISBN 978-1-4503-8833-7. Available from DOI: 10.1145/3407023.3409192.
52. EUROBSDCON. *Paul Vixie talks about DNS over HTTPS* [online]. 2019 [visited on 2021-04-30]. Available from: <https://www.youtube.com/watch?v=ZxTdEEuyxHU&t=2491s>.
53. CHU, Jerry; JAIN, Arvind; CHENG, Yuchung; RADHAKRISHNAN, Sivasankar. *TCP Fast Open* [online] [visited on 2021-05-02]. Available from: <https://tools.ietf.org/html/rfc7413>.
54. *The Sad Story of TCP Fast Open* [The Squeeze] [online]. 2019-04-11 [visited on 2021-05-02]. Available from: <https://squeeze.isobar.com/2019/04/11/the-sad-story-of-tcp-fast-open/>. Section: Ideas.

55. DECCIO, Casey; DAVIS, Jacob. DNS privacy in practice and preparation. In: *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies* [online]. Orlando Florida: ACM, 2019, pp. 138–143 [visited on 2021-05-02]. ISBN 978-1-4503-6998-5. Available from DOI: 10.1145/3359989.3365435.
56. *devconf-2014-kernel-networking-walkthrough-16-638.jpg (638×479)* [online] [visited on 2021-05-02]. Available from: <https://image.slidesharecdn.com/devconf2014-kernelnetworkingwalkthrough-140304102610-phpapp01/95/devconf-2014-kernel-networking-walkthrough-16-638.jpg?cb=1393929597>.
57. *zmap/zmap* [online]. The ZMap Project, 2021 [visited on 2021-05-15]. Available from: <https://github.com/zmap/zmap>. original-date: 2013-01-23T01:30:09Z.
58. LU, Chaoyi; LIU, Baojun; LI, Zhou; HAO, Shuang; DUAN, Haixin; ZHANG, Mingming; LENG, Chunying; LIU, Ying; ZHANG, Zaifeng; WU, Jianping. An End-to-End, Large-Scale Measurement of DNS-over-Encryption: How Far Have We Come? In: *Proceedings of the Internet Measurement Conference* [online]. Amsterdam Netherlands: ACM, 2019, pp. 22–35 [visited on 2021-05-09]. ISBN 978-1-4503-6948-0. Available from DOI: 10.1145/3355369.3355580.
59. *curl/curl* [GitHub] [online] [visited on 2021-05-09]. Available from: <https://github.com/curl/curl>.
60. *Welcome to Knot DNS's documentation! — Knot DNS 2.6.9 documentation* [online] [visited on 2021-05-09]. Available from: <https://www.knot-dns.cz/docs/2.6/html/index.html>.
61. WEILER, S.; BLACKA, D. *Clarifications and Implementation Notes for DNS Security (DNSSEC)* [online] [visited on 2021-05-09]. Available from: <https://datatracker.ietf.org/doc/html/rfc6840#section-5.7>.
62. *Determine if the IP Address of a DNS Resolver in Route 53 Supports the ECS Extension* [Amazon Web Services, Inc.] [online] [visited on 2021-05-09]. Available from: <https://aws.amazon.com/premiumsupport/knowledge-center/route-53-find-ecs-support-dns-resolver/>.
63. WOOLF, S.; CONRAD, D. *Requirements for a Mechanism Identifying a Name Server Instance* [online]. 2007-06 [visited on 2021-05-09]. Available from: <https://www.ietf.org/rfc/rfc4892.txt>.
64. *OpenSSL Cookbook: Chapter 1. OpenSSL Command Line* [online] [visited on 2021-05-09]. Available from: <https://www.feistyduck.com/library/openssl-cookbook/online/ch-openssl.html>.

65. Doporučení v oblasti kryptografických prostředků [online]. 2018, p. 8 [visited on 2021-05-16]. Available from: [https://www.nukib.cz/download/uredni\\_deska/Kryptograficke\\_prostredky\\_doporuceni\\_v1.0.pdf](https://www.nukib.cz/download/uredni_deska/Kryptograficke_prostredky_doporuceni_v1.0.pdf).
66. MCKAY, Kerry A; COOPER, David A. *Guidelines for the selection, configuration, and use of Transport Layer Security (TLS) implementations* [online]. Gaithersburg, MD, 2019-08 [visited on 2021-05-16]. NIST SP 800-52r2. National Institute of Standards and Technology. Available from DOI: 10.6028/NIST.SP.800-52r2.
67. ALI, Al Imem. Comparison and Evaluation of Digital Signature Schemes Employed in NDN Network. *International Journal of Embedded Systems and Applications* [online]. 2015, vol. 5, no. 2, pp. 15–29 [visited on 2021-05-16]. ISSN 18395171. Available from DOI: 10.5121/ijesa.2015.5202.
68. *Lucky Thirteen: Breaking the TLS and DTLS Record Protocols* [online] [visited on 2021-05-16]. Available from: <http://www.isg.rhul.ac.uk/tls/Lucky13.html>.
69. *SHattered* [online] [visited on 2021-05-16]. Available from: <https://shattered.io/>.