Vulnerabilities and Possible Defenses of the Domain Name System

***Abstract*—Possible attacks like Distributed Denial of Service attacks, Cache Poisoning or Name Chaining, ID Guessing and Query Prediction, as well as Betrayal By Trusted Server attacks all threaten a critical base infrastructure of the Internet, the Domain Name System. This paper discusses how the afore mentioned attacks are carried out on the current structure of the Domain Name System as well as concepts that intend to guard it now and against future attack concepts such as DNSSEC or DNS-over-HTTPS. While this paper informs about possible concepts that diminish threats, it also highlights the limitations of those concepts as well as the success of other implementations and concepts. Furthermore, the practicability of various solution concepts is discussed.**

I. Introduction

The Domain Name System (DNS) is a crucial part of the modern Internet. It translates domain names into IP-Addresses and stores the corresponding names and addresses in a tree-like structure. As there are more than 400 billion responses [1] every day the DNS can be considered critical base infrastructure of the Internet. Used in many protocols such as Internet Message Access Protocol (IMAP), File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP) or Secure Shell (SSH), DNS's availability is closely related to availability of other services [2].

Therefore, constant service and integrity is required for the Internet to work properly. If larger parts of the DNS were compromised or not available due to an attack many Internet services and most of the 4.54 billion active Internet users [3] would be affected. This is especially true in regions where working Internet connections are essential for everyday life.

Even though DNS is designed to be decentralized the internal tree-like structure is often identified as a week spot. The theoretical disadvantages caused by such a scenario range from longer loading times to severe privacy issues such as identity theft or fraud on a minor to massive scale or unavailable services. Internet outages like this could result in millions of dollars of lost turnover for big e-commerce sites [4].

To prevent attackers performing such acts several proposals have been made within recent years. *DNS-over-HTTPS* (previously DNS-over-TLS) as a measure against eavesdropping currently gains a lot of popularity especially since many daily Internet users are increasingly more cautious about their privacy. The Mozilla Foundation / Mozilla Corporation enables this security measure by default in their Firefox web browser in the US as of early 2020 [5].

The Chromium Projects also tests DNS-over-HTTPS in the Chrome web browser as of versions 79 and 83 with a slightly different approach. Chrome checks if the currently by the OS defined DNS provider offers DNS-over-HTTPS and upgrades the request if the provider does. If the provider does not support DNS-over-HTTPS Chrome sticks to a standard DNS request [6].

This paper evaluates some of the afore mentioned proposals reviewing their abilities as well as their practicability or current state of implementation. While the following section will provide a general overview of how DNS works, section three examines well known weak spots by means of some selected examples. Using these examples section four compares proposals to enhance the resilience and security of DNS. Section five then lists some existing implementations and their success followed by a short conclusion in section six.

II. Structure and Mechanisms of DNS

The Domain Name System is a decentralized and scalable system that provides translation like services in a network such as the Internet. It is used by almost every network accessing application and thus plays a vital role in most networks.

A request towards the Domain Name System can be compared to an address lookup. While most people do not remember actual IP addresses most people know the Uniform Resource Locator (URL) of the desired online service like "example.com" which directs to 93.184.216.34. A URL is a text based address which DNS can translate into an IP address that the user's device can use, to connect to the desired server.

The communication in between the client and the DNS server is managed with DNS messages. These messages always consist of five fields called header, question, answer, authority and additional (see figure 1). The header always consists of 12 bytes, two of which are reserved for the ID, a unique number generated by the requesting party. It also includes one bit to distinguish queries from responses, 15 more bits for administrative purposes and four times 16 bits that determine the count of entries in the corresponding message fields (see figure 2).

The length of the other fields may vary but will never exceed 512 bytes in total. In case the response is overly long the recipient will be notified by the truncate flag (TC) set in the header and can repeat the request if not satisfied with the response [7].

This human readable address is divided from right to left. The address "example.com" for example is split into the so called labels "example" and "com". Each label is then translated and the request is forwarded to the DNS server responsible for the respective zone. The relation of a label and the corresponding address can be described as a table. The

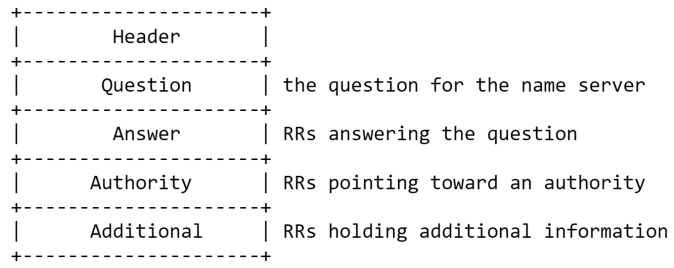

Fig. 1. The structure of a DNS Message as specified by IETF, copied from RFC1035 [7].

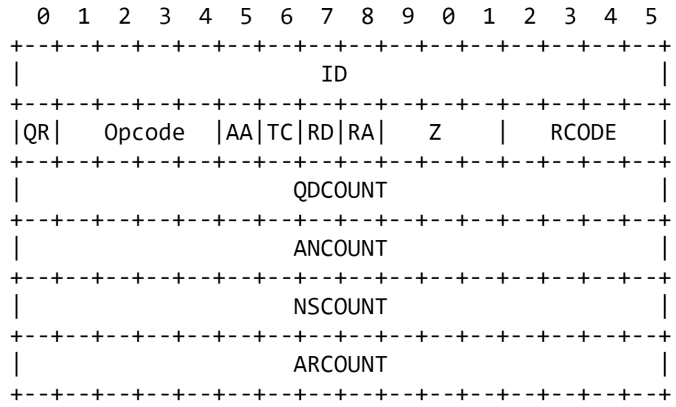

Fig. 2. The structure of a DNS Message Header as specified by IETF, copied from RFC1035 [7].

entries, referred to as resource records (RR), often redirect to another table instead of an IP address which leads to the tree like structure the DNS consists of. The tree's root starts of with a top level domain (TLD) like "com" or "edu". From there the paths split up to different nodes which are managed by other DNS servers. This pattern repeats with every node in the tree. The path from the root node to a leave of the tree then represents a Fully Qualified Domain Name (FQDN) such as "www.example.com". An example of a resolve tree is represented by figure 3. Most people will recognize "www" which is a web server specific subdomain and short for "world wide web". Other subdomains frequently used are "mail" or "imap" for mail servers or "ftp" for mail servers. Such resource records contain, other than the address they redirect to, fields like Time To Live (TTL) that specify how long the record stays valid. Resulting from the desire to reduce the workload of DNS servers it is usual practice to cache resource records. This allows the number of steps taken to resolve an address to be reduced. Such cached records always consist of a TTL which requires the record to be dismissed or refreshed once the TTL expires.

For large bandwidth applications such as video streaming it is impractical to deliver the desired content from far away servers. DNS therefore also provides the option to direct to the server closest to the requesting party. A request from Vienna might be resolved differently than a request from Seattle. This

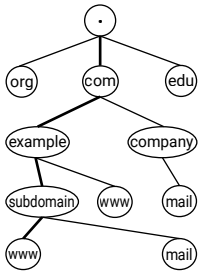

Fig. 3. The tree of a "www.subdomain.example.com" resolve

helps the service to operate with lower latency which increases user satisfaction and service operability.

III. Possible Security Risks and Theoretical Attacks

There are many possible scenarios that pose threats to the DNS. While some attacks may target the availability, others try to compromise the data integrity or resource records. The resulting security and/or privacy issues vary and depend on the intent of the compromising party. Even though one might argue that DNS records are public anyway and therefore question what sensible information might be leaked, it is important to know that the combination of requesting IP address and other information such as timestamps, detailed information about the connection and the requesting user may be deduced. This information could be used for targeted advertising, analysis of browsing behavior or even surveillance.

An attack that targets system availability is the *Distributed Denial of Service Attack* (DDoS). A DDoS Attack floods the victim server with traffic from multiple attacking clients, often using thousands of computers, mostly personal computers hijacked with malware. This keeps legitimate requests from coming through and makes the attacked server practically useless, hence denial of service. An attack on a DNS Server may not only lead to failing requests resolves on the attacked server but also affect other servers that forward to the attacked one. This flaw may render large clusters of DNS Servers unavailable since often other servers in the cluster depend on the attacked one to resolve part of their requests.

When a domain or the server it points to changes, the corresponding resource record in the DNS structure changes as well. In order to direct every user according to the updated record, DNS servers exchange their records. This means address changes propagate over time instead of instantly. If servers that direct to a zone are not available due to an ongoing DDoS attack, updates of resource records can not be pushed to other zones. This leads to services not directly connected to the DNS being unavailable and therefore major security risks. Take the case of servers that host security updates for operating systems. In case their address cannot be resolved, necessary security updates cannot be installed.

Another possible attack concept is cache poisoning which aims to change the data in the RR with malicious intent. One can argue that *cache poisoning* is the most prominent

and dangerous attack [8]. It aims to return altered and thus incorrect data (e.g another IP address) to the resolver. The resolver then connects to a different server than originally intended. Possible identity theft by not trustworthy services or censorship as done by the Chinese government [9] for sure are examples to why the security of DNS constantly needs to be enhanced. An attack of this type could be carried out via name chaining. In combination with cache poisoning this means, that an attacker responds to a victim's DNS request with one or more Resource Records that lead to different servers while keeping the name, the victim wanted to resolve. An attacker might lead the victim's request to a DNS server of the attacker's choice. Since the responses can be constructed like the attacker needs them to look like, false information cannot just be placed in the answer section of a response, but also in the authority section. This way, the malicious information has a better chance to get past the victim's resolver's defenses. In that case, the attacker can place malicious information in the victim's cache which may lead to further attacks [10]. Phishing also is a common consequence of cache poisoning. Being redirected to a server chosen by the attacker allows the possibility of forged web pages. Tapped user inputs such as usernames and passwords pose a major security risk as online services such as online banking or online administrative services become more and more popular.

ID Guessing and Query Prediction may not be the most effective or efficient solution to send false DNS responses to a victim however, it is fairly easy for an attacker to generate response packets that match a victim's DNS request header. This way, an attacker may send false responses without the victim's notice. Furthermore, an attacker does not need to be on the same network as the victim. The possible combinations the attacker has to guess in order to falsify a DNS request can be reduced from 2^{32} to 2^{16} since the port the victim uses can be determined by analyzing the victim's previous traffic. The only parameter the attacker needs to guess is the ID field value in the DNS response header, since the port a DNS server uses is known, QNAMEs and QTYPEs can be predicted using previous traffic as well. The ID field value may also be predicted based on the victim's previous traffic, however an attacker might just brute-force the ID since 2^{16} possibilities are not enough to make a brute-force attempt too time-consuming and therefore not worth trying. If the attacker guesses the ID right, a false response can be sent and lead the victim to a completely different destination without the victim's notice [10].

Another possible security risk for the Domain Name System is called *betrayal by trusted server*. An example for betrayal by trusted server would be standard DNS servers implemented by Internet Service Providers (ISP) through their routers or the DHCP protocol. These DNS servers may be trustable at first, when the ISP uses the IP address to have a standard DNS server defined in their routers. However, if the ISP changes the address of the DNS server or the DNS server goes offline because the ISP does not keep on supporting it's own DNS service, the IP address defined in the ISPs routers might not be changed. This would lead to a potentially grave security risk. If an attacker knows that the DNS server used in the ISPs routers will not be supported in the future, the attacker could set up a false DNS server using the originally trustable DNS server's IP address that would lead requests by victims to different sites chosen by the attacker. This would leave potential for phishing attacks maybe without users noticing [10]. Although this scenario is highly unlikely since many mistakes by the ISP are required in order to lead to this situation, it shows how fragile DNS is when it comes to changes in DNS server addresses as well as the importance of standard DNS servers with static IP addresses that will not change in the future.

Betrayal by trusted server may be carried out in various ways. One way could be the one described above without malicious intent. Another variant of the attack with malicious intent would consist of a third-party DNS services, seemingly trustworthy. If these DNS services are being used by an ISP to handle DNS requests from their clients, user data and browsing behavior could be collected and sold by the utilized DNS services. Furthermore, the third-party DNS service might as well send malicious DNS responses that lead to phishing attacks. This means that trustworthy seeming third-party DNS services can carry out betrayal by trusted server attacks. In this case the ISP helped to create the attack by accident using the third-party DNS provider. [10].

IV. Possible Solutions and Impact of Implementation

Because DNS has been used since the beginnings of the Internet as we know it today, current changes have to be backwards compatible to support software and services that require the implementation that currently exists. Furthermore, the changes need to be usable by all clients as fast as possible to guarantee the success of the implementation as a new standard. If there were multiple standards at once, a global platform like the Internet is today would not be achievable with newer implementations.

A typical defense against DDoS attacks used to be and still is to simply provide such an abundance of resources that even medium sized attacks do not lead to a major outage. Proposals for alternatives to this effective but yet expensive defense mechanism have been made in large numbers. Many of them target the avoidance of big outages by improved caching (for example longer TTLs or more frequent refreshes) of resource records. This could be achieved with longer TTLs for every resource record [11]. However, other problems occur with this proposition. After a resource record's time to live exceeds, the RR gets either updated or dismissed. If there is an update to a RR, it may just refresh the TTL of the record. However, the old RR may possibly point to a wrong destination until every zone receives the updated record. Other methods may cause bigger headers in DNS messages and therefore more overall traffic. The required hardware adaptions however would still be cheaper than providing an abundance of hardware resources.

Cache Poisoning as a privacy issue has been addressed relatively early in the history of DNS. The commonly used defense

strategy is the deployment of Domain Name System Security Extensions (DNSSEC). DNSSEC adds a digital signature to the RR which can be validated by the requesting party. It is therefore ensured that the delivered record redirects to the desired address and does not contain any compromised data. The IETF suggested an initial version of DNSSEC in March 1999 in RFC 2535 [12]. However by 2012 only about 0.02% of zones had deployed DNSSEC [13]. By early 2020 about "25% of the world's [user's resolvers]" [14] use DNSSEC for validating their DNS responses. However, the little use of DNSSEC is not caused by DNS servers that do not add a signature to their response packets. Many of the clients simply do not check for a valid signature in the responses they receive [14]. This shows that DNSSEC is still taking time for mainstream availability although it protects against many risks of malicious DNS responses.

Recent growth in the use of DNSSEC is due to changes in the Asian regions. With many affected users by regional changes, improvements in Asia have a big impact on the global usage of DNSSEC [14].

The original version of DNSSEC proofed to be too complex and therefore too resource intensive for fast widespread adoption [?]. It took a few iterations of improvements and the announcement of severe vulnerabilities of the standard DNS implementation by Dan Kaminsky, the so called Kaminsky Bug, for DNSSEC deployment to gain traction. This shows how long it takes innovation to be elaborated and wide spread in internet applications. While the usage of DNSSEC has certainly increased since it's proposal, the deployment of DNSSEC should further be prioritized as every potential poisoning poses a thread to user privacy and a possibility for spoofing attempts.

A possibility to protect users against ID guessing and query prediction would be to require signature checking of all responses to DNS requests. This way, only valid responses from real DNS servers would be accepted [10]. To eliminate risks of these kind of attacks, every application that communicates with DNS servers would need to implement a check like this. This poses a problem for older applications that are not supported, or applications that do not receive updates anymore. In favor of backwards compatibility this solution cannot be enforced. Nonetheless, DNS servers could send a signature with every response which has to be checked by applications whenever possible.

The signature checks proposed with DNSSEC could, under certain conditions, help to diminish risks for betrayal by trusted server attacks as well. In case of the first example, given in Section III of this paper, a signature check would allow responses from the false DNS server setup with the same IP address as the originally trustworthy DNS server. This would protect the victim of a possible phishing attack [10]. However in the second example, the DNS server hosted by the third-party company contracted by the victim's ISP does not change. In that case, signature checks would not fail since the server did not change. This example shows the limitations of DNSSEC. Since software cannot be used to determine if a DNS server is trustworthy or not, further work is required in order to save users and providers from misleading DNS services. One possibility could be a regulatory standard for third-party DNS services. It could be mandatory for third-party DNS providers to get checked by independent institutions repeatedly. This way, users and ISPs would have a standard which can be used to determine the trustability of a DNS provider. Overall, this may lead to less malicious DNS services since they could not receive the standard due to their bad practices. Unfortunately, to the best of our knowledge, a regulatory standard like this one have not been proposed yet.

With growing privacy awareness among users and the wish to circumnavigate authoritative censorship, DNS-over-HTTPS (the de facto successor to DNS-over-TLS) is a recently introduced method to encrypt the DNS traffic. Though still in testing, it has proven not to reduce performance by a substantial amount [15]. This renders third parties unable to read the DNS messages contents. Encryption is the only way to ensure the confidentiality of these sent messages. As per default there is no encryption implemented in the DNS everyone with physical access to a shared or transit network may be reading messages and therefore gathering potentially sensitive information.

V. Current Implementations and their Success

The following description of current DNS-over-HTTPS implementations are also depicted in figure 4.

Currently the Mozilla Foundation / Mozilla Corporation enables DNS-over-HTTPS by default in their Firefox web browser in the US as of early 2020. The way this works is by redirecting every DNS request to a DNS-over-HTTPS capable DNS server. This way DNS requests and browsing behavior is hidden from other users on the client's public or local network and the client's ISP. Therefore, the browser enables Internet usage with more privacy. [5]

The usage of DNS-over-HTTPS limits services and organizations that protect their users from malicious sites on the Internet by using their user's DNS requests. Firefox acknowledges this by offering a DNS-over-HTTPS bypass. This way an application that needs access to DNS requests can continue operating with minor changes, that bypass the default setting in Firefox. [5]

One might argue that the DNS provider that Firefox uses is still able to collect the users data and browsing behavior. However, Firefox redirects requests only to selected DNS operators that they have contracts with, that legally bind the operators to adhere to Firefox's Trusted Recursive Resolver policy. This policy requires the providers to only collect personal user data that is required for service operation. Furthermore, data that could identify a user has to be removed after 24 hours after the request, the data was collected from, is resolved. Other data that cannot identify users can be kept for more than 24 hours [16].

The Chromium Projects chose a slightly different approach by not redirecting every request to a designated DNS provider. Instead, every request to the DNS service defined by the

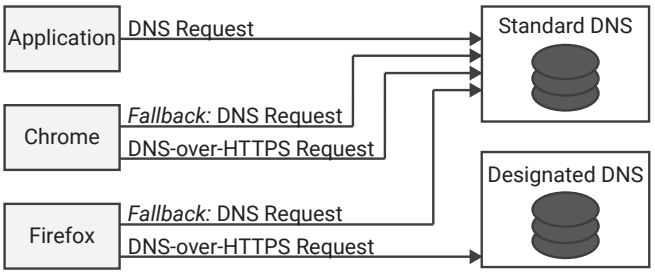

Fig. 4. Different approaches applications and browsers [5] [6] take with DNS-over-HTTPS

operating system the browser is running on gets checked for a DNS-over-HTTPS alternative by the same DNS operator. This way, the request is using a different protocol but the user's DNS provider remains unchanged. This approach is called "Auto-Upgrade" because the protocol get's upgraded if possible [6].

"Auto-Upgrade" however, does not keep the DNS providers from collecting user data and browsing behavior. Users DNS requests would still be hidden from public or local networks but not necessarily from ISPs if the ISP's DNS server is the default for a network with devices that do not have a different DNS server defined by their operating system or users choice.

Both approaches have fallback mechanisms if the use of DNS-over-HTTPS fails to resolve a DNS request. Firefox uses a normal DNS request to the standard DNS provider defined by the operating system, if the designated DNS provider cannot resolve the DNS-over-HTTPS request. This way, all requests should be resolved as usual and when possible over HTTPS without disrupting the users daily browsing experience [5]. Chrome also uses a normal DNS request as well as the standard DNS provider if "Auto-Upgrade" leads to a failed DNS-over-HTTPS resolve.

The development teams of Chrome and Firefox are both currently testing their approaches. Firefox with it's US user base and Chrome as an experiment in version 83 [6] to a at this point in time unknown extend.

If both browsers implemented DNS-over-HTTPS as the default DNS request protocol, about 77.36% of desktop browser users (as of March 2020 [17]) and more than 61.96% of mobile browser users (as of February 2020 [18]) would benefit from more privacy and security.

With this major marketshare (see figure 5) in the desktop and mobile web browser market, the Mozilla Foundation / Mozilla Corporation and The Chromium Projects have the power to set new standards which is essential given that the Internet and it's wide variety of applications have to keep working non-stop. Unlike DNSSEC that still has not reached a validation quote of 50% [14] more than 20 years after it's proposal [12], DNS-over-HTTPS has the chance to be adopted far quicker due to the way it is distributed.

Defenses against DDoS attacks have theoretically alternatives to outpacing the attackers with more hardware. However the alternatives that involve improved caching methods are

Desktop Browser	Marketshare	Mobile Browser	Marketshare
Chrome	68.11%	Chrome	61.96%
Firefox	9.25%	Firefox	unknown
Others	22.64%	Others	38.04%

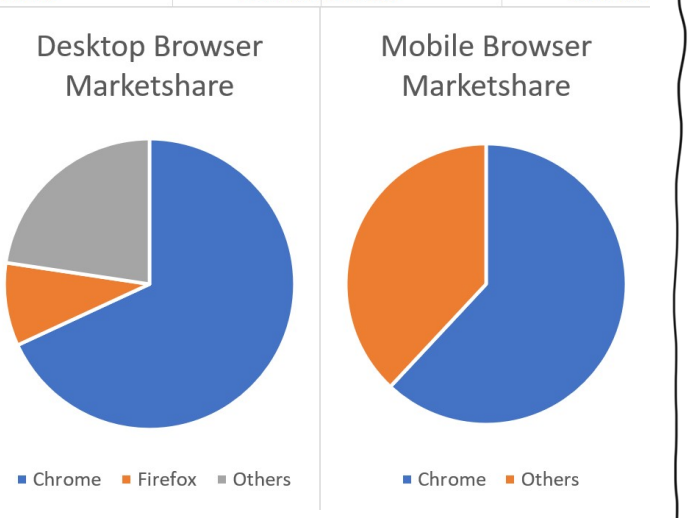

Fig. 5. The marketshare of the Chrome and Firefox web browser applications on desktop (as of March 2020 [17]) and mobile (as of February 2020 [18]) devices

creating new problems like longer update times for resource records or a large increase in traffic caused by larger DNS message headers [11]. It is conceivable that these solutions are not necessarily the most effective but alternatives that should always be considered.

VI. Conclusion

DNS is one of the most important infrastructures of the modern Internet. More secure approaches to DNS have been researched, identified and developed for years. Many of them are promising and some of them are already in place to make the Internet and it's applications more secure and reliable.

DNS-over-HTTPS and DNSSEC fixed a lot of issues the original implementation of the DNS seemed to have. However, many risks and threats persist and will persist until newer security measures are developed and adopted. DNS still is used as a tool for censorship against it's original purpose and none of the solutions and proposals can fix that. While DNSSEC is not the solution to all of DNS's security flaws, it could path the way to a more secure future of the Internet. However, for that to happen, DNSSEC needs to be adopted a lot faster than it happened before. DNSSEC is a complex system that requires careful programming and more resources as bigger response packets are send and validation operations have to be performed. These and other characteristics of DNSSEC in turn imply risks, such as easier DDoS attacks due to higher workload. Confidentiality is a factor completely unaddressed by DNSSEC. Hence a combination with other security measures is required. Additionally the prevention of DDoS attacks on DNS servers currently do not have any practicable alternatives.

A good example for wide adoption of a security measure are the current tests for DNS-over-HTTPS by the Mozilla Foundation / Mozilla Corporation and The Chromium Projects. As both of their browsers are testing to adopt the new security standard it could help to limit man in the middle attacks to a small amount, given their dominance in the desktop and mobile web browser market. Initiatives like these help to adopt new standards on a large scale in a short amount of time.

More regulatory standards that could help ISPs and other users identify trustworthy DNS service providers could help to secure DNS by minimizing the risks of malicious third-party DNS services.

In conclusion, there are many proposals that would improve DNS security and protect privacy for users. However, developing and distributing all these solutions is often not practical or takes a long time. Securing the DNS is an ongoing effort that takes engagement of services like web browsers with a large user base to take effect for most of the Internet's users.

REFERENCES

[1] Y. Gu, "Google Public DNS and Location-Sensitive DNS Responses," https://webmasters.googleblog.com/2014/12/google-public-dns-and-location.html, accessed: 2020-04-10.

[2] L. Wei-Min, C. Lu-Ying, and L. Zhen-Ming, "Alleviating the impact of DNS DDoS attacks," in *2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing*, vol. 1. IEEE, 2010, pp. 240–243.

[3] Statista, "Global digital population as of January 2020," https://www.statista.com/statistics/617136/digital-population-worldwide/, accessed: 2020-04-11.

[4] E. Nygren, R. K. Sitaraman, and J. Sun, "The akamai network: A platform for high-performance internet applications," *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 3, p. 2–19, Aug. 2010. [Online]. Available: https://doi.org/10.1145/1842733.1842736

[5] Mozilla, "Firefox DNS over HTTPS (DoH) FAQs," https://support.mozilla.org/en-US/kb/dns-over-https-doh-faqs, accessed: 2020-05-05.

[6] The Chromium Projects, "DNS over HTTPS (aka DoH)," https://www.chromium.org/developers/dns-over-https, accessed: 2020-05-05.

[7] P. Mockapetris, "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION," https://www.ietf.org/rfc/rfc1035.txt, pp. 25–26, accessed: 2020-05-05.

[8] S. Son and V. Shmatikov, "The Hitchhiker's Guide to DNS Cache Poisoning," in *Security and Privacy in Communication Networks*, S. Jajodia and J. Zhou, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 466–483.

[9] A. Klein, H. Shulman, and M. Waidner, "Internet-wide study of dns cache injections," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[10] D. Atkins, IHTFP Consulting, R. Austein, "Threat Analysis of the Domain Name System (DNS)," https://tools.ietf.org/html/rfc3833, pp. 4–8, accessed: 2020-05-05.

[11] V. Pappas, D. Massey, and L. Zhang, "Enhancing dns resilience against denial of service attacks," in *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*, 2007, pp. 450–459.

[12] I. D. Eastlake, "Domain name system security extensions," https://tools.ietf.org/html/rfc2535, accessed: 2020-05-05.

[13] A. Herzberg and H. Shulman, "Antidotes for dns poisoning by off-path adversaries," in *2012 Seventh International Conference on Availability, Reliability and Security*, 2012, pp. 262–267.

[14] Geoff Huston, "DNSSEC validation revisited," https://blog.apnic.net/2020/03/02/dnssec-validation-revisited/, accessed: 2020-05-06.

[15] T. Böttger, F. Cuadrado, G. Antichi, E. L. a. Fernandes, G. Tyson, I. Castro, and S. Uhlig, "An empirical study of the cost of dns-over-https," in *Proceedings of the Internet Measurement Conference*, ser. IMC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 15–21. [Online]. Available: https://doi.org/10.1145/3355369.3355575

[16] Mozilla, "Security/DOH-resolver-policy," https://wiki.mozilla.org/Security/DOH-resolver-policy, accessed: 2020-05-05.

[17] Statista, "Global market share held by leading desktop internet browsers from January 2015 to March 2020," https://www.statista.com/statistics/544400/market-share-of-internet-browsers-desktop/, accessed: 2020-05-05.

[18] Statista, "Market share held by leading mobile internet browsers worldwide from January 2012 to February 2020," https://www.statista.com/statistics/263517/market-share-held-by-mobile-internet-browsers-worldwide/, accessed: 2020-05-05.