Mitigation method for random subdomain attacks to authoritative DNS servers by traffic control

Hikaru Ichise
Information Technology Center, Oita University
Oita, Japan, ichise-hikaru@oita-u.ac.jp

Yong Jin
Center for Information Infrastructure, Institute of Science Tokyo
Tokyo, Japan, yongj@cii.isct.ac.jp

Minoru Ikebe
Faculty of Science and Technology, Oita University
Oita, Japan, minoru@oita-u.ac.jp

Katsuyoshi Iida
Information Initiative Center, Hokkaido University
Sapporo, Japan, iida@iic.hokudai.ac.jp

***Abstract*—DDoS(Distributed Denial of Services) attack has been one of the most critical cybersecurity threats on the Internet. DDoS attacks send a lot of requests to specific servers, such as DNS (Domain Name System), SMTP (Simple Mail Transfer Protocol), Web servers, and so on. Eventually these target servers will be forced to stop their services due to the significant high load caused by the DDoS attacks. Random subdomain attack is one of the DDoS attacks, which aims at the authoritative DNS servers.**

A huge amount of DNS queries for randomly generated subdomain names will be sent to the target authoritative DNS server via a lot of DNS full-service resolvers in a short time period. Consequently, the process of handling non-existent subdomain names will cause a rapid increase in the resource usage of the authoritative server, such as CPU and memory. Thus, when random subdomain attacks occurred on an authoritative server, the Internet users cannot use the Internet services appropriately.

In this paper, we propose a mitigation method for the random subdomain attacks by rate limit and traffic control. The proposed method can mitigate the significant increase in the workload and help the authoritative DNS server keep the name resolution service for the legitimate users. We constructed a local network environment and evaluated the proposed method. The results confirmed that the proposed method could mitigate approximately 3.5 times CPU load under pseudo-random subdomain attacks.

***Index Terms*—DDoS attacks, Random subdomain attacks, Authoritative server, Rate limit, traffic control, DNS cookie**

I. Introduction

DDoS (Distributed Denial of Service) attack is one of the most critical cyberattacks on the Internet. Several reports have indicated that the number of DDoS attacks has increased drastically nowadays [1]. Moreover, the attackers perform DDoS attacks using multiple protocols, including TCP, UDP, HTTP, DNS(Domain Name System), and so on. For example, when a large number of Internet users, including the DDoS attacks, access the website, the legitimate Internet users are not able to use the web content from the web server because of the high load on the Web server. Since many web accesses include legitimate usage, network administrators are difficult to restrict communication easily.

DDoS attacks disrupt a variety of services provided to legitimate users by target servers. Figure 1 illustrates the overview of DDoS attacks. Firstly, a computer is infected with bot malware by email or web browsing (a bot-infected computer). Next, the bot-infected computer detects and is commanded from the C&C (Command and Control) server (arrow 1) and sends a huge amount of requests to the target server (arrow 2). As a result, legitimate computers are denied from being served by the target server due to its overload. In other words, it is difficult for the target server to provide various services. Some researchers reported that there are a variety of types of DDoS attacks [2]–[4] and random subdomain attack is one of them targeted at the Domain Name System (DNS) [5]. In this paper, we mainly focus on detecting and mitigating the random subdomain attacks.

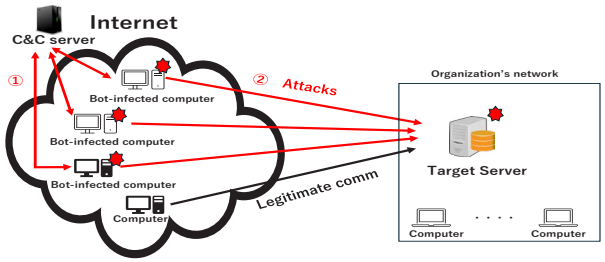

Fig. 1. The overview of DDoS attack

DNS is one of the most important infrastructures on the Internet [6], [7] for translating domain names to IP addresses (name resolution) on the Internet. Figure 2 shows a general name resolution process using DNS full-service resolver and authoritative DNS servers. Firstly, an end terminal sends a DNS query to the DNS full-service resolver. Next, the DNS full-service resolver performs name resolution by iteratively querying the corresponding authoritative DNS servers in the Internet. Then, the DNS full-service resolver replies with the DNS response to the end terminal. It should be noted that the DNS full-service resolver receives NXDOMAIN response from the authoritative DNS server when the domain name has not been registered in the corresponding authoritative DNS server. In addition, once the DNS full-service resolver receives an IP address corresponding to the domain name from the authoritative DNS server, the DNS full-service resolver will

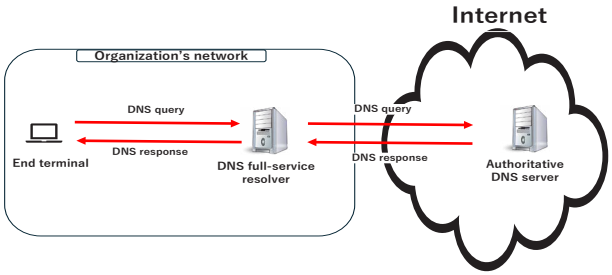

Fig. 2. Name Resolution Process

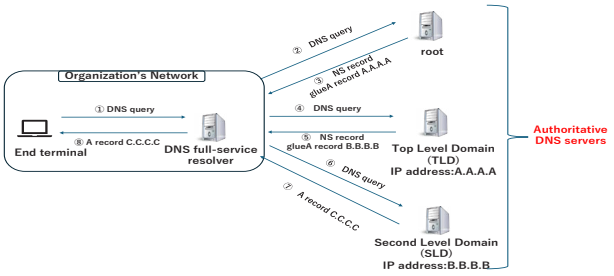

Fig. 3. The simple name resolution process

cache it temporarily. The authoritative DNS server receives DNS queries and sends responses from numerous DNS full-service resolvers of each organization's network constantly.

In random subdomain attacks, bot-infected computers generate random and unique subdomain names and send a huge amount of DNS queries for those random subdomain names to the DNS full-service resolver of the organization's network. Then, the DNS full-service resolver conducts iterative DNS queries for those requests to the target authoritative DNS server. As a result, the workload of the target authoritative DNS server will increase rapidly. Consequently, legitimate computers cannot perform domain name resolutions using the target authoritative DNS server so that various applications, including mail and Web, cannot be used. Therefore, it is necessary to protect the authoritative DNS servers from random subdomain attacks.

However, it is not easy to detect random subdomain attacks on the authoritative server at the early stage, since there has been no system for checking the random subdomain names of DNS queries. In this paper, we propose a mitigation method for random subdomain attacks on the authoritative DNS server.

II. Random Subdomain Attacks and Related Work

As mentioned in the Introduction, the objective of this study is to detect and mitigate random subdomain attacks on authoritative DNS servers. In this section, we introduce the detailed procedure of the random subdomain attack and describe some related research.

A. Random Subdomain Attacks

Figure 3 shows a simple name resolution process. First, an end terminal sends a DNS query to the DNS full-service resolver in the organization's network (arrow 1). Next, the DNS full-service resolver sends the DNS query to the authoritative DNS servers iteratively from the root to the Second Level Domain(SLD) (arrows from 2 to 7). Finally, the end terminal obtains the IP address from the full-service resolver.

Figure 4 depicts an overview of random subdomain attack. In a random subdomain attack, a computer infected by a bot program (bot-infected computer) generates a huge amount and unique subdomain names and sends the DNS queries for those subdomain names at a significantly high rate to the DNS full-service resolver in the organization's network. In case of no DNS record has been registered in the authoritative server, the authoritative server replies with an NXDOMAIN (non-existent domain) name [8] DNS response.

In general, there is a rate limit for the NXDOMAIN response in an authoritative DNS server. When the limit is exceeded, the authoritative DNS server will reply with a DNS response indicating the DNS full-service resolver query using TCP protocol. Then the DNS full-service resolver will send the same query using TCP protocol, which is called "TCP fallback".

The result is that the workload of the target authoritative DNS server will be significantly increased rapidly, including CPU and memory, since communication between the DNS full-service resolver and the authoritative DNS server becomes TCP protocol instead of UDP protocol. Eventually, the authoritative DNS server will stop the name resolution service due to the high workload. Thus, it is essential to detect and mitigate the random subdomain attacks on authoritative DNS servers.

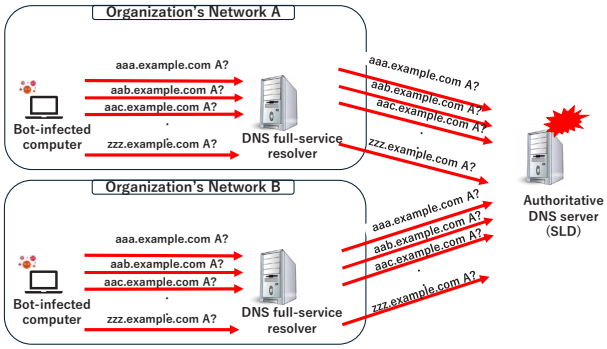

Fig. 4. The overview of Random Subdomain attack

B. Related Work

There has been some researches in the literature regarding random subdomain attacks.

In [9], the authors analyzed 595 times of random subdomain attacks which occurred in 2018. They classified three types of random subdomain attacks through the analysis: subdomain attacks on the authoritative DNS server, subdomain attacks with DNS full-service resolver bypass, subdomain attacks targeting the DNS full-service resolver. Specifically, the authors reported that most of the attacks occupy to target the authoritative DNS server. However, the paper lacked a clear description of the countermeasures to the random subdomain attacks.

In [10], the authors proposed a subdomain name server, which stores the URLs in the TXT record of the authoritative

DNS server. DNS full-service resolver queries the TXT record from the authoritative DNS server. DNS full-service resolver checks the subdomain name from the subdomain name server with the URL of the TXT record. Then, the DNS full-service resolver queries the IP address of the subdomain name to the authoritative DNS server again. However, this system may cause high latency in the name resolution process.

The methods proposed in the existing researches, are difficult to use as a practical mitigation method for random subdomain attacks in a real network environment. Therefore, in this paper, we propose a mitigation method for random subdomain attacks on authoritative DNS servers.

III. Proposed System

As explained in Sect. II-A, the targets of random subdomain attacks are authoritative DNS servers. In this section, we first introduce the response rate limit and DNS cookies, which are options commonly used in authoritative DNS servers. Then, we explain the system architecture regarding the mitigation of random subdomain attacks in detail.

A. Response rate limit and DNS cookies

Response rate limit [11] is one of the special features in a DNS server for controlling the number of UDP DNS responses per second. The objective is to regulate the number of UDP DNS responses per second to prevent DNS amplification attacks or DDoS attacks on DNS servers. This feature can be set since BIND version 9.10. Based on the response rate limit configurations.

DNS cookie [12] provides a security feature in a DNS server, which aims to be consistent between the end terminal and the DNS server in the name resolution process. This feature is effective for the DNS amplification attacks and cache poisoning attacks. Once the feature is valid, the response rate limit is effective and the DNS full-service resolver will not perform TCP fallback. However, DNS cookie have increased the network load for the large data volume. Thus, the name resolution process can enhance the network performance by the invalidity of the send-cookie parameter of the DNS cookie.

By using the response rate limit and the DNS cookie in the proposed system, DNS queries will be regulated as UDP packets from random subdomain attacks effectively. The proposed system is based on the following four observations.

(1) In a legitimate name resolution process, an authoritative DNS server receives DNS queries and replies responses moderately.

(2) An authoritative DNS server receives a huge amount of DNS queries for the subdomain names and replies NXDOMAIN responses under a pseudo random subdomain attacks since there are no resource records registered for those subdomain names.

(3) There is a rate limit set on the authoritative DNS server and when the response rate exceeds the limit, the authoritative DNS server will reply a response indicating use TCP protocol to the DNS full-service resolver.

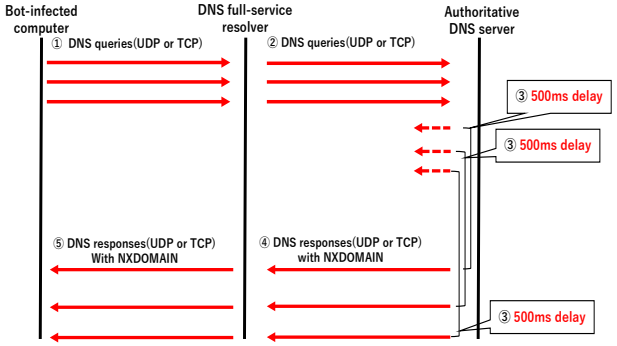

Fig. 5. Overview of workflow in proposed system

(4) Then, the DNS full-service resolver sends a huge amount of DNS queries for the subdomain names using TCP protocol to the target authoritative DNS server and stop the name resolution service.

The key idea is to mitigate the DNS query rate of random subdomain attacks by controlling the traffic between the authoritative DNS server and the DNS full-service resolver. Specifically, a piece of delay will be set on the DNS response from the target authoritative DNS server to the DNS full-service resolver. As a result, the DNS full-service resolver will wait for the response with delay and send the next query so that the number of DNS queries sent to the target authoritative DNS server per second can be shortened.

Figure 5 shows the basic workflow of the proposed system. First, the bot-infected computer sends a huge amount of DNS queries using UDP to the DNS full-service resolver. The DNS full-service resolver regulates UDP packets to the authoritative server.

Then, the authoritative DNS server spends time sending DNS responses intentionally, which is intended to reduce the CPU consumption. Considering the comprehensive features for the response rate limit and the time lag in DNS traffic, we use the `tc` (Traffic Control) command [13] to realize the feature.

B. System Architecture

On the authoritative DNS server, we used the following parameters for the response rate limit and DNS cookie :

- Slip: The number of UDP responses until TCP fallback.
- Response per second: The number of responses that are sent by UDP per second. The communication between the DNS full-service resolver and the authoritative DNS server uses TCP protocol when the number exceeds this value.
- NXDOMAIN per second: The number of NXDOMAIN responses sent using UDP per second.
- Send-cookie: Configure whether the DNS full-service resolver sends a DNS query with a DNS cookie to the authoritative server.
- Answer-cookie: Configure whether the DNS full-service resolver receives a DNS response with a DNS cookie from the authoritative server.

In addition to the above parameters, we use the `tc` command to add delay for DNS responses from the authoritative

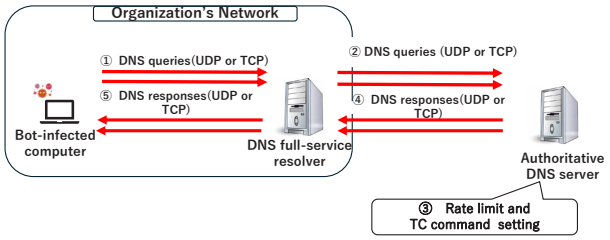

Fig. 6. System Architecture

DNS server to the DNS full-service resolver. `tc` command [13] is a tool for reproducing the network incidents, containing network delay, packet loss, network throughput, and so on. Based on the configuration, the proposed system can reduce the CPU load of the authoritative DNS servers under random subdomain attacks. Figure 6 depicts a brief system architecture of our proposed system using the rate limit and `tc` command. We simply explain the procedure of the proposed system.

1) A bot-infected computer, such as a DGA (Domain Generation Algorithm) based bot program, generates random subdomain names on the computer in advance. Then, the bot-infected computer sends a huge amount of DNS queries to perform the name resolution of the random subdomain names to the DNS full-service resolver.
2) The DNS full-service resolver sends those DNS queries to the authoritative server because there is no cache stored in the DNS full-service resolver.
3) The authoritative DNS server sets the rate limit and the `tc` command to regulate the number of DNS queries with the UDP protocol and to control the DNS traffic to add delay. This is intended to reduce the CPU load of the authoritative DNS server.
4) The authoritative DNS server spends the time based on the `tc` command configuration so that the DNS full-service resolver receives the DNS responses with NXDOMAIN with a delay.
5) The DNS full-service resolver replies with NXDOMAIN responses to the bot-infected computer.

Our objective is to mitigate the resource consumption of the authoritative DNS server under random subdomain attacks. Therefore, with these procedures, our proposed system can protect the authoritative DNS server from random subdomain attacks. It should be noted that there is no delay in the proposed system for the legitimate name resolution, which is the case of sending the DNS queries for the subdomain names with DNS resource records registered in the authoritative server. Because in this case, the DNS response from the authoritative DNS server is not NXDOMAIN.

IV. Implementation and Evaluation

Based on the system architecture, we constructed a local network environment and DNS servers in order to implement a prototype system.

TABLE I
SPECIFICATIONS OF TESTBED ENVIRONMENT

Component	OS	Software	CPU	RAM
Host machine A and B	AlmaLinux	KVM	4 core	8GB
DNS full-service resolver (KVM)	AlmaLinux	BIND	1 core	2GB
Authoritative DNS server (KVM)	AlmaLinux	BIND	1 core	2GB
computer	Mac	dnsperf	6 cores	32GB

TABLE II
PARAMETER SETTING IN BIND FOR THE IMPLEMENTATION

Feature	parameter name	parameter value
Response Rate Limit	slip	1
	responses-per-second	30
	nxdomains-per-second	30
DNS cookie	send-cookie	no
	answer-cookie	yes

A. Implementation

In order to perform the random subdomain attacks, we prepared a file containing 10K random and unique subdomain names created by a Python program [14] and none of them have DNS resource records registered in the authoritative DNS server. For DNS full-service resolver and authoritative DNS servers, we used BIND. Table I shows the specifications of the machines used in the implementation.

Figure 7 illustrates the experimental network environment. The Mac computer was installed with a DNSperf tool [15] which can send DNS queries using the file to a DNS full-service resolver for emulating a random subdomain attack. The option "-Q" of the DNSperf regulates the maximum number of DNS queries per second. We constructed two physical machines, Host server A and B. In Host server B, we created three virtual machines as the authoritative DNS server in the same network segment using KVM, a virtualization machine. In Host server A, we created a virtual machine as the DNS full-service resolver. On the target authoritative DNS server (the SLD authoritative DNS server), we added delay using `tc` command and set the response rate limit as well as DNS cookie as shown in Table II. We add 500ms delay on the traffic from the target authoritative DNS server to the DNS full-service resolver.

B. Feature Evaluation

First, we measured the resource usage of the SLD authoritative DNS server under a pseudo random subdomain attack.

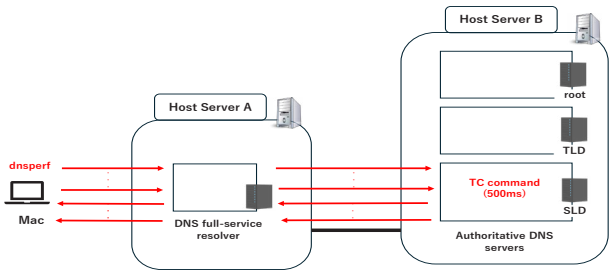

Fig. 7. Network Environment

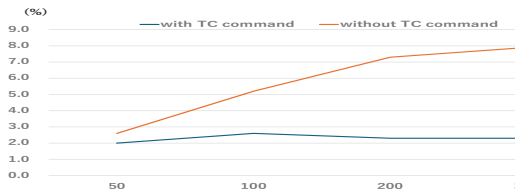

Fig. 8. CPU load rate in SLD authoritative DNS server

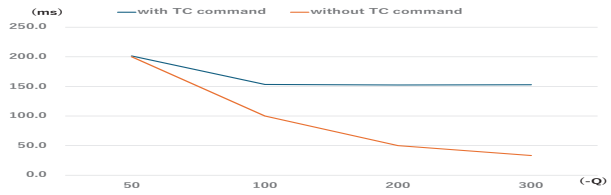

Fig. 9. Execution time in DNSperf

Particularly, we checked three values: CPU consumption, run time in the DNSperf, and queries per second in the DNSperf.

Figure 8 shows the CPU load in the SLD authoritative DNS server. Orange and blue lines show the CPU load without and with the delay respectively. As shown, in the case of without delay, the CPU load increases along with the "-Q" value in the DNSperf. On the other hand, when we set the delay, the CPU load decreased obviously. In particular, when the "-Q" value is 300, the number of our proposed system is 3.5 times lower than the number of the no prevention method. This means that the SLD authoritative DNS server mitigates the random subdomain attacks by the proposed system.

Furthermore, as shown in Figure 9, the DNSperf in Mac shortens the run time without the delay along with the "-Q" value in the DNSperf. On the other hand, the execution time of the DNSPerf with the delay takes 150 milliseconds to receive the DNS responses of 10K subdomain names. Figure 10 shows the queries per second in the DNSperf. The proposed system can suppress the number of DNS queries per second, even if the authoritative DNS server receives queries at a high rate.

As a result, the proposed system caused the TCP fallback in communication between the DNS full-service resolver and the authoritative DNS server in all cases. However, based on the result of the CPU load in the SLD authoritative DNS server, the run time in the DNSperf, and the queries per second in the DNSperf, we confirmed that the proposed system is effective in mitigating the pseudo random subdomain attacks on the authoritative DNS server.

V. Conclusion

The main objective of the proposed method is to mitigate random subdomain attacks on an authoritative DNS server by setting response rate limit and appropriate traffic control. In this paper, we designed and described the system architecture of the proposed method. Based on the proposed method, we implemented a prototype system by using an authoritative DNS server and constructed an experimental network environment. Then we evaluated the features of the

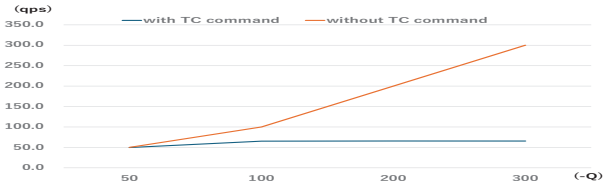

Fig. 10. Queries per second in DNSperf

proposed method in the experimental network environment. The results confirmed that the proposed method can mitigate approximately 3.5 times CPU load under a pseudo random subdomain attack on the authoritative DNS server.

In future work, we plan to extend the proposed method by adding the traffic delay for the DNS NOXDOMAIN responses only and evaluations in a real network environment.

References

[1] S. Dummer, D. Birchard, S. Rath, and B. Van Nice, "DDoS Attack Trends in 2024 Signify That Sophistication Overshadows Size," (online) available from https://www.akamai.com/blog/security/ddos-attack-trends-2024-signify-sophistication-overshadows-size, April 07, 2025.

[2] I. A. Valdovinos, J. A. Pérez-Díaz, K. R. Choo, and J. F. Botero,, "Emerging DDoS attack detection and mitigation strategies in software-defined networks: Taxonomy, challenges and future directions," Journal of Network and Computer Applications, Volume 187, 2021, 103093, pp. 1084-8045.

[3] G. Somani, M. S. Gaur, D. Sanghi, M. Conti, R. Buyya,, "DDoS attacks in cloud computing: Issues, taxonomy, and future directions," Computer Communications, Volume 107, 2017, pp. 30-48.

[4] I. Sharafaldin, A. H. Lashkari, S. Hakak and A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," 2019 International Carnahan Conference on Security Technology (ICCST), Chennai, India, 2019, pp. 1-8, doi: 10.1109/CCST.2019.8888419.

[5] Akamai , "What Are Pseudo-Random Subdomain Attacks?," https://www.akamai.com/glossary/what-are-pseudo-random-subdomain-attacks, Accessed on Aug. 16, 2025.

[6] P. Mockapetris, "DOMAIN NAMES - CONCEPTS AND FACILITIES," IETF RFC1034, Nov. 1987.

[7] P. Mockapetris, "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION," IETF RFC1035, Nov. 1987.

[8] Y. Iuchi, Y. Jin, H. Ichise, K. Iida, and Y. Takai, "Detection and Blocking of DGA-based Bot Infected Computers by Monitoring NXDOMAIN Responses," 2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), New York, NY, USA, 2020, pp. 82-87.

[9] H. Griffioen and C. Doerr, "Taxonomy and Adversarial Strategies of Random Subdomain Attacks," 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Canary Islands, Spain, 2019, pp. 1-5.

[10] L. Pan, R. Qiu, and M. Yang, "ASDWL: Mitigating DNS Random Subdomain Attacks for Second Level Domain," 2024 International Conference on Smart Applications, Communications and Networking (SmartNets), Harrisonburg, VA, USA, 2024, pp. 1-4.

[11] J. Dickinson, S. Dickinson, R. Bellis, A. Mankin, and D. Wessels, "DNS Transport over TCP - Implementation Requirements," IETF RFC7766, March. 2016.

[12] D. Eastlake, and M. Andrews, "Domain Name System (DNS) Cookies," IETF RFC7873, May. 2016.

[13] M. P. Stanic, "TC—Traffic control," Linux QOS Control Tool, 2001, https://arvanta.net/mps/linux-tc.pdf, Accessed on Aug. 16, 2025.

[14] "Python," https://www.python.org, Accessed on Aug. 16, 2025.

[15] DNS-OARC, "dnsperf," https://www.dns-oarc.net/tools/dnsperf, Accessed on Aug. 16, 2025..