Proceedings of the International Conference on Intelligent Computing and Control Systems (ICICCS 2019)
IEEE Xplore Part Number: CFP19K34-ART; ISBN: 978-1-5386-8113-8

A Survey on DNS Security Issues and Mitigation Techniques

Anju Ramdas
Center for Cybersecurity Systems & Networks
Amrita Vishwa Vidyapeetham
Amritapuri, India

Ramakrishnan Muthukrishnan
Least Authority TFA GmbH
Berlin

***Abstract*—The Domain Name System (DNS) is the backbone of the internet. It is a distributed hierarchical database which stores resource records like A, MX, AAAA, CNAME. The whole DNS is classified into three layers - root, top-level domain (TLD) and authoritative DNS servers. Each level has its own responsibility to resolve certain categories of domain names. It is very difficult for us to memorize the IP address of each site which we need to visit. In this case, the DNS comes into rescue to figure out the corresponding IP address a domain points to. In the current world, the internet is an inevitable part of our life and DNS is the soul of the internet. Due to this reason, DNS is a major attack target like amplification attack, cache poisoning attack, DNS hijacking, NXDomain attack and Phantom domain attack. These attacks could create a serious security threat to internet users. Threats can be a simple redirection to potentially stealing user credentials. Even though different mitigation techniques are available, the threat still exists. In this paper, we present our survey of the existing research and its shortcomings on securing the DNS. We have also introduced a novel idea which uses blockchain technology to validate the response sent by the DNS servers.**

Keywords—DNS security, cache poisoning, hijacking, DNSSEC

I. INTRODUCTION

As internet usage is increasing tremendously, so is the number of websites. To visit any website, we need the IP address of the web server which hosts the site. Searching the website using the IP address is not an easy task. Moreover, the IP address of a site will be changing. This will create more overhead to the user.

This is the time where DNS comes into play. It is a hierarchical database which is mainly used to map the domain name to the corresponding IP address. Along with the IP address, the DNS server stores other records like MX, AAAA, and CNAME. DNS is an application-layer protocol which enables a user to query the distributed database. The DNS uses UDP as its underlying protocol and uses port 53. DNS consists of a large number of servers which is organized in a hierarchical manner. These servers are distributed across the world. Basically, there are three layers of DNS servers - root DNS server, TLD and authoritative DNS servers. No single DNS servers hold all mappings. There are 13 root DNS servers distributed across the world. This server stores the resource records of the TLD servers. There are many TLD servers like for GOV, EDU, COM, MIL, ORG, NET. Each TLD server will return the resource record for the corresponding authoritative DNS server which maintains the domain. In the final step, the authoritative DNS server will return the IP address of the website. There is one more class of DNS server called local DNS server. This server is maintained by the Internet Service Provider (ISP). The DNS lookup can be performed in two ways - recursive and iterative. In the recursive technique, the root, TLD and authoritative DNS servers communicate with each other to figure out the IP address and the final answer will be sent from the root server to the local DNS server. Whereas in the iterative method, the local server will only do all the communication to figure out the IP address. To minimize the lookup time, each server cache the response. So whenever a request came for a domain name, it will check its cache. But if the stored IP address is fake, it will lead to cache poisoning attack.

The DNS lookup is done using DNS query. The query and the response in the DNS are not encrypted and no authentication is done before communication. This is the main reason why DNS is vulnerable to many attacks. Popular attacks in DNS are amplification attack, DNS cache poisoning, DNS hijacking, NXDomain attack, and phantom domain attack. In this paper, we present the popular attacks happened on DNS and the defence mechanism & approaches for hardening the DNS security. The main contribution of this research study are:

- Review of DNS protection mechanisms.
- Proposal for introducing the blockchain technology to validate the response sent by the DNS servers.

The rest of this paper is organized as: Section II provides our observation of each of the significant research works towards securing the DNS. Section III presents a summary of our future work, which essentially consists of the blockchain technology to validate the DNS response. Section IV consists of the conclusion. Table I shows the summary of some DNS protection mechanism.

Authorized licensed use limited to: National Institute of Technology Karnataka Surathkal. Downloaded on September 21,2024 at 18:37:30 UTC from IEEE Xplore. Restrictions apply.

II. DNS Threats and Defence Approaches

When the client sends the DNS query, the Questions section in the query contains information about the query. In the DNS response, the Answers section contains the resource record for the domain name in the Questions section.

The Identification flag in the query contains the Query ID. The client will accept the DNS response only when 1) the response arrives on the same UDP port as sent by the client 2) the Query ID matches the pending query and the Question section matches the Question in the pending query. In the traditional DNS, the Query ID is incremented for each DNS request. As a result, the attacker could easily guess the Query ID and will succeed in sending the DNS response that matches with all the requirement the client machine is looking for in a response. In this case, the Answers section will have the IP address of the website created by the attacker. Then the attacker could steal login credentials and also will poison the cache of the server. This is DNS cache poisoning. This attack affects all the servers which are connected to the poisoned server.

Lack of origin authentication [13] is another serious issue in DNS. The client system will blindly trust the DNS response even without authenticating the source or validating the response. This issue creates room for DNS hijacking attack. Privacy is another issue affected by DNS. Both the DNS query and response are not encrypted. As a result, ISP or anyone who could intercept the message could identify the user and the website that the user is visiting.

Further, in this section, we will discuss our perspectives and observations of the prominent research works and publications that have contributed to secure the DNS.

A. Blockchain Backed DNSSEC

Gourley [1] and the co-authors proposed an extension to the DNSSEC [14]. The proposed technique prevents availability issues and minimizes the number of request for signature verification in DNSSEC. Instead of the two signing keys, they have introduced a new key called Combined Signing Key (CSK). This reduces key management complexity. To reduce the number of request for signature verification, they make use of X.509 certificates. This certificate is linked to a domain name and a CSK. The X.509 certificates are stored inside X.509 blockchain [2]. The blockchain is maintained by the Certificate Authority (CA). Whenever the recursive resolver receives the A record of a server, it will first validate it using the X.509 certificate. If it is valid, the resolver will forward the A record to the client. Otherwise, it will discard the A record.

B. CGuard

CGuard [3] protects the recursive resolver from cache poisoning attack. In cache poisoning attack the attacker will send a lot of responses to the targeted server. In the proposed technique, a server will detect the attack by recording the number of responses it received for a query. When the number of responses for a query is greater than a threshold value, it will operate using high confidence channel. This high confidence channel can be DNS lookup using TCP, DNSSEC, UDP double query (UDQ).

C. E-DNSSEC

The proposed technique uses E-DNSSEC (Encrypted DNSSEC) [4] for preserving the privacy of DNSSEC queries. In the traditional DNSSEC, both the query and the response are not encrypted. As a result, anyone can read the query and tamper it according to the way they want. In the proposed technique, the recursive resolver will encrypt the query using the RSA algorithm and inserts it inside the request. The authoritative server will extract the query from the request and decrypts it. All other processing is done using the DNSSEC protocol. The proposed technique will only allow an authentic user to see the query.

D. Oblivious DNS

In the traditional DNS, the recursive resolver can see the query, the response and the client's IP address. This creates huge privacy issues for the client. For preserving the privacy of the client, Schmitt and the co-authors introduced Oblivious DNS (ODNS) [5]. The proposed technique introduced two components – ODNS stub and ODNS resolver. ODNS stub sits between the client browser and the recursive resolver. It will obfuscate the DNS query sent from the client. Thus the recursive resolver cannot see the domain name the client requests. ODNS resolver sits between the recursive resolver and the authoritative DNS server. It obfuscates the client's IP address when the recursive resolver sends the request to the authoritative DNS server. ODNS prevents an attacker from linking a DNS query with the client IP address that issued the query.

E. DNS Guard

DNS Guard [6] is proposed to prevent IPv6 DNS reconnaissance attack [15]. DNS guard is implemented in the Intrusion Detection System (IDS). DNS guard consists of 4 components - packet inspector, packet detection engine, packet decision maker and packet DB. The packet stream from the network is fed to the packet inspector. It uses rules which are configured to filter out the specified packets and forward them to the packet detection engine. The packet detection engine will consult the packet DB for the historical record of that query. If there is a previous record, then it will check for DNS reconnaissance patterns. If the pattern is found, then it will save this captured packet detail in the packet DB and sends this information to the Packet decision maker. The Packet decision maker will drop the packet and sends an alert to the network administrator. In the paper itself, they have mentioned that this technique is not compatible with all IDS like Snort.

F. A Large Scale Analysis of DNS Water Torture Attack

Luo [7] and the co-authors conducted a large scale analysis of the water torture attack [10] and found that the DNS water torture attack is larger in number and more random than that of disposable domains. So they introduced a technique to mitigate this issue at the local DNS server. They have analyzed first

Authorized licensed use limited to: National Institute of Technology Karnataka Surathkal. Downloaded on September 21,2024 at 18:37:30 UTC from IEEE Xplore. Restrictions apply.

10,000 DNS queries and calculated $n_{q,}$ d_d and d_c. n_q is the number queries, d_d is the number of distinct domains and d_c is the number of distinct clients. They concluded that when a water torture attack happens, the value of d_d divided by n_q or d_c divided by n_q is greater than 70%. Then they have identified the victim second level domain (SLD) so that the network administrator could block the outside traffic to the victim domain.

G. Mitigation Process for DNS Flood Attacks

Mahjabin and the co-author proposed a technique [8] to prevent DNS Flood attack. In the proposed technique two lists are maintained to handle the traffic. LIST SERVER is used to maintain the list of servers available to handle the traffic and LIST TRAFFIC is used to store the traffic arrived at the server. When traffic arrives at the server, the system checks the server's list for an available server and directs the traffic to the server. When a particular amount of time is collapsed, the system will stop accepting any new traffic and only serve already arrived traffic. In this way, the system can protect itself from a large volume of queries.

III. Future Work

The most researched problem in DNS is to secure the DNS from well-known attacks like cache poisoning attack and DNS hijacking. The DNS hijacking has now become a common weapon used by cyber-terrorist and hackers. Even though a lot of mitigation techniques exist against the hijacking attack, the threat still remains. In DNS hijacking, the victim computer will be compromised by overriding a computer's TCP/IP configuration to point at an attack controlled DNS server, through modifying the behaviour of a trusted DNS server so that it does not comply with DNS protocol or tamper the resource record by stealing the login credentials of the server.

Our proposal is an extension to the existing DNS. This proposal uses blockchain technology. Blockchain has many applications like in the field of healthcare, IoT [9], finance, supply chain, smart contracts. For each field, there are different types of blockchain. For example, Ethereum [11] is used for smart contract and Bitcoin [12] is used for finance. In our proposal, blockchain is used to store the domain name to IP address mapping. The blockchain is maintained by the DNS servers. After doing the DNS lookup, the DNS server will send the DNS response. At the client side, the resolver will use a similar technique like simplified payment verification (SPV) to calculate the hash of the IP address which it received and to check if the hash is in the block header. If it is there, then the resolver will send the IP address to the client. Else it will discard it. So just by using the Merkle tree root, the resolver will be able to check if the IP address is stored in the blockchain. Since blockchain is tamper-proof, the client can trust the validation process.

IV. Conclusion

The DNS is an application layer protocol used by all internet users. As a result, it is a target to a large number of attacks. These attacks not only affects the DNS servers but also the internet itself. So it is very important to figure out the ways to secure DNS. In this paper, we discussed some existing approaches which can be used to protect DNS. We also discussed our future work towards securing the DNS using blockchain technology.

References

[1] S. Gourley and H. Tewari, "Blockchain Backed DNSSEC," Business Information Systems Workshops Lecture Notes in Business Information Processing, pp. 173–184, 2019.

[2] H. Tewari, A. Hughes, S. Weber, and T. Barry, "X509Cloud — Framework for a ubiquitous PKI," MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM), 2017.

[3] S. Y. Chau, O. Chowdhury, V. Gonsalves, H. Ge, W. Yang, S. Fahmy, and N. Li, "Adaptive Deterrence of DNS Cache Poisoning," Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering Security and Privacy in Communication Networks, pp. 171–191, 2018.

[4] K. Chetioui, G. Orhanou, and S. El Hajji, "New Protocol E-DNSSEC to Enhance DNSSEC Security", IJ Network Security, 20(1), pp.19-24, 2017.

[5] P. Schmitt, A. Edmundson and N. Feamster, "Oblivious DNS: Practical Privacy for DNS Queries", arXiv preprint arXiv: 1806.00276, 2018.

[6] Q. Hu, M. R. Asghar, and N. Brownlee, "Measuring IPv6 DNS Reconnaissance Attacks and Preventing Them Using DNS Guard," 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2018.

[7] X. Luo, L. Wang, Z. Xu, K. Chen, J. Yang, and T. Tian, "A Large Scale Analysis of DNS Water Torture Attack," Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence - CSAI 18, 2018.

[8] T. Mahjabin and Y. Xiao, "Mitigation Process for DNS Flood Attacks," 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), 2019.

[9] S. Sankaran, S. Sanju, and K. Achuthan, "Towards Realistic Energy Profiling of Blockchains for Securing Internet of Things," 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), 2018.

[10] Y. Takeuchi, T. Yoshida, R. Kobayashi, M. Kato, and H. Kishimoto, "Detection of the DNS Water Torture Attack by Analyzing Features of the Subdomain Name," Journal of Information Processing, vol. 24, no. 5, pp. 793–801, 2016.

[11] C. Dannen, "Dapp Deployment," *Introducing Ethereum and Solidity*, pp. 149–157, 2017.

[12] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, 2008.

[13] S. Ariyapperuma and C. J. Mitchell, "Security vulnerabilities in DNS and DNSSEC," The Second International Conference on Availability, Reliability and Security (ARES07), 2007.

[14] O. Kolkman and R. Gieben, "DNSSEC Operational Practices," 2006.

[15] D. Zagar and K. Grgic, "IPv6 Security Threats and Possible Solutions," 2006 World Automation Congress, 2006.

Authorized licensed use limited to: National Institute of Technology Karnataka Surathkal. Downloaded on September 21,2024 at 18:37:30 UTC from IEEE Xplore. Restrictions apply.

TABLE I. SUMMARY OF SOME DNS PROTECTION MECHANISMS

TECHNIQUE	SUMMARY	LIMITATION	PREVENTS
Blockchain Backed DNSSEC	Proposed a new technique as an extension to the DNSSEC. The recursive resolver will validate the DNS response by using X.509 certificate of the domain. X.509 certificates are stored in the blockchain.	If an attacker is successful in creating X.509 certificate for the malicious site, then this technique cannot prevent hijacking attack.	DNS hijacking attack
CGuard	Switches to high confidence channel when the number of responses received at the server is greater than a threshold value. High confidence channel can be DNS lookup using techniques like LTS, PR-TCP.	Figuring out the accurate threshold value is difficult. If the local name server detected the attack, it need to resend the query based on any high confidence channel technique.	Cache poisoning attack
E-DNSSEC	Introduced a new technique called Encrypted DNSSEC for preserving the integrity of the DNS query. The local DNS server encrypts the query and inserts it in the DNS request. The authoritative DNS servers receives the DNS request and decrypts it.	Already DNSSEC have complex key management. Adding this extra encryption and decryption may create overhead and can cause delay in processing.	Data leakage
Oblivious DNS	Introduced Oblivious DNS (ODNS) for preserving the privacy of the client. It obfuscates the DNS query sent from the client and the client's IP address from upper levels of the DNS hierarchy. ODNS prevents an attacker from linking a DNS query with the client IP address that issued the query.	ODNS components knows both IP address and domain name queried by the client. So it can be compromised to link the DNS query with the IP address of the client.	Data leakage
DNS Guard	DNS Guard is implemented in IDS. DNS guard consists of four components - Packet inspector, packet detection engine, packet decision maker and packet DB.	DNSG is not compatible with all IDS like Snort (mentioned in the paper).	IPv6 DNS reconnaissance attack
A Large Scale Analysis of DNS Water Torture Attack	Analyzed first 10,000 DNS queries. Among them, they found out the victim second level domain (SLD). Then the network administrator could block the outside traffic to the victim query.	The DNS queries collected for analysis is done only at the ISP level. So it doesn't contain the whole traffic in the water torture attack. So the result may be incorrect or can't predict that it can stop the attack completely.	Water torture attack
Mitigating Process for DNS Flood Attacks	Maintains LIST SERVER and LIST TRAFFIC to keep track of number of available servers and number of traffic respectively. When traffic arrives at the server, the system checks the server's list for an available server and directs the traffic to the server. These steps are followed till there is no remaining traffic in the list. After some time, the system will stop accepting new traffic and process existing traffic.	While the system stops to accept the traffic, legitimate user may receive delayed response.	DNS flood attack

Authorized licensed use limited to: National Institute of Technology Karnataka Surathkal. Downloaded on September 21,2024 at 18:37:30 UTC from IEEE Xplore. Restrictions apply.