DNS Defense and Security Investigation

Zuyu Wang
School of Engineering and Application Science
George Washington University

Caozhiyuan Huang
School of Engineering and Application Science
George Washington University

Zifan Qu
School of Engineering and Application Science
George Washington University

Abstract—**The domain name system (DNS) is an important infrastructure of the Internet, and its main function is to provide domain name resolution services. With the development of the Internet, DNS has also been given other application functions, such as DKMI (domain keys identified mail) Standards , load balancing , domain name blocking , etc. Most Internet applications need to rely on DNS to work properly. Once the DNS system is attacked, the entire Internet will be severely affected.**

The DNS protocol currently in use mainly follows the RFC 1034 and RFC 1035 specifications. In order to make up for the lack of information verification in the DNS protocol, RFC 4034 introduced a DNSSEC enhancement scheme based on PKI technology. However, the security threat of the DNS system has not been reduced. Various DNS security incidents in recent years have not only caused serious economic losses to the attacked organizations and companies, but also have a serious impact on the stability of the entire Internet. With the emergence of some new attack methods, DNS security environment has become more severe, such as: using Internet of Things devices to launch DDoS (distributed denial-of-attack) attacks on the DNS system, using DNS messages to analyze user privacy information, using DNS to establish tunnels Attack .

In response to various security threats to DNS, as of December 2017, there are as many as 256 RFC documents related to DNS. Scholars have proposed many novel ideas for DNS security issues. This article addresses DNS security threats in recent years . The research work related to protection is analyzed and reviewed, and the types and causes of DNS threats are summarized, and the security enhancement methods and research progress of DNS in terms of protocol, system implementation, diagnostic monitoring, privacy protection, and architecture are introduced.

Section 1 of this article introduces the security status of DNS. Section 2 classifies DNS threats and summarizes the root causes of the vulnerability of the DNS system. Section 3 lists various security enhancement solutions for the vulnerability of the DNS system, and compares and analyzes them. Section 4 proposes hot spots and prospects for DNS security research work. Finally, a brief summary is given.

1. DNS security status

With the continuous development of network technology, the technology for attacking DNS has become more abundant and the means has become more complicated. The American Coleman Parkes company investigated the security status of DNS systems from a total of 1,000 organizations in North America, Asia Pacific, and Europe and found that , In 2017, 76% of organizations were attacked by DNS. Among these attacks, malware attacks accounted for 35%, DDoS attacks accounted for 32%, cache poisoning accounted for 23%, DNS tunneling accounted for 22%, and zero-day vulnerability attacks accounted for 19 %. More than 90% of malware uses the DNS protocol to keep in touch with the malware's command and control (CC) center to obtain attack commands, download software updates, and obtain private information. DDoS attacks are becoming more and more serious. As it becomes more complex, attackers use a wide range of technical means, from basic methods (e.g. amplification/forwarding, flooding) to highly complex attacks involving botnets and chain reactions. These attacks may come from internal or external DNS servers. According to a survey report published by Arbor Network, 84% of reflection and amplification attacks use the DNS protocol, which is the highest proportion of all survey protocols. In addition, the report also shows that the DNS server is a DDoS attack The primary goal, 78% of DDoS attacks attack the application layer services of DNS.

Attacks on DNS are profitable, and commercial interests are driven to intensify attacks. Some attackers use DNS servers to disrupt corporate services, damage corporate reputation, and cause user loss. For example, in October 2016, Dyn domain name service provider suffered a large-scale DDoS After the attack, Dyn lost 8% of its domain name customers. DNS attacks can also cause key data leakage and economic losses. According to EfficientIP's survey report, one-third of the 1,000 companies and organizations surveyed DNS attack data is stolen. 16% of these data are user sensitive information and 15% are intellectual property information. In addition, DNS attacks will cause the victim company to cost 2 million US dollars in economic losses every year.

DNS attack technology is becoming more and more abundant, but the detection technology to deal with it is relatively scarce. According to a Cisco research report, an attacker can control more than 100,000 IoT devices within 24 hours through malware. These IoT devices can launch large-scale DDoS attacks. Because these malware are stored in the memory of the victim device and are automatically cleared after the device is shut down, it is difficult to collect samples of malicious programs. The zero-day vulnerability of the server software makes detection difficult. Increased, 83% of companies did not install security patches in time, resulting in further intensified attack effects.

2. Security Threat Analysis

2.1. DNS Hierarchy and Vulnerability Analysis

DNS is mainly composed of three parts, namely:

1) domain name space and resource record, including the tree structure of the namespace and the data associated with the name;

2) the name server , a server program containing domain tree structure information and setting information;
resolver, which responds to requests and obtains query results from the name server.

DNS usually provides two domain name resolution methods, namely: recursive query and Iterative query.
In general, a host query to a local domain name server uses a recursive query; the so-called recursive query is if the local domain name server sent by the host does not recognize the IP address of the domain name being requested, the local domain name server will continue to send query request messages as a DNS client to another root domain name server (that is, to continue querying for request messages).

Iterative query query from the local domain name server to the root domain name server; iterative query characteristics: When the root domain name server gets an iterative query request response received by the local domain name server, it either provides the IP address to be queried, or informs the local server: ”Which domain name server should you query next?”

After introduced the architecture of DNS, it is necessary to know the internet domain structure before going through the process of domine name resolve.

The Internet uses a hierarchical tree structure naming system for naming the Internet because of the huge number of Internet users; any host or router connecting to the Internet has a special hierarchical name i.e. a domain name (domain name).); ’domain’ here is a manageable division in the namespace; grammatically speaking, each domain name consists of a label set, and each label is divided by a dot (decimal point)

The structure of domin name is shown in Figure 2.

It can be seen from Figure 1 and Figure 2 that the DNS servers on the Internet are also arranged in a hierarchical manner. Each domain name server only has jurisdiction over a part of the domain name system. Domain name servers can be classified into four distinct groups, based on the position played by the domain name server:

1) Root Domain Name Server: The highest-level domain name server and also the most relevant domain name server; all root domain name servers are familiar with the domain names and IP addresses of all top-level domain name servers; no matter which local domain name server it is it is mandatory for every domain name server on the Internet.

2) Top-level domain name server: Responsible for handling the registered second-level domain names on the top-level domain name list

3) Authority domain name server: responsible for a ”zone” domain name server;

4) Local domain name server: The local server does not belong to the hierarchical structure of the domain name server in the figure below, but it is very important to the domain name system; when a host sends a DNS query request, the query request message is sent to the local domain name server;

Based on the architecture of domain name and DNS server, the domain name resolve process has six steps:

1) The recipient makes a proposal for domain name resolution and submits the request to the central domain name server.

2) When the request is handled by the local domain name server, it first searches the local cache. If a record object occurs, the local domain name server immediately returns the output of the query.

3) If no such record remains in the local cache, the local domain name server sends the request directly to the root domain name server, and then the root domain name server returns the address of the primary domain name server of the queried domain (root subdomain) to the local domain name server.

4) The local server sends a request to the domain name server that has been returned from the previous stage, and then the server that approves the request sends its own cache and returns the address of the applicable lower domain name

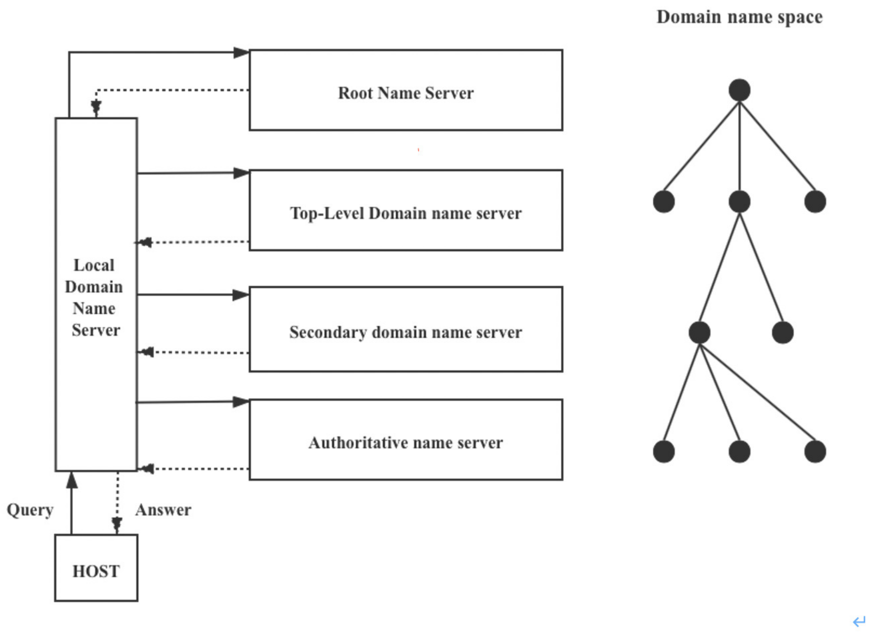

Domain name space
Root Name Server
Top-Level Domain name server
Secondary domain name server
Authoritative name server
Local Domain Name Server
Query
Answer
HOST

Figure 1. DNS Architecture.

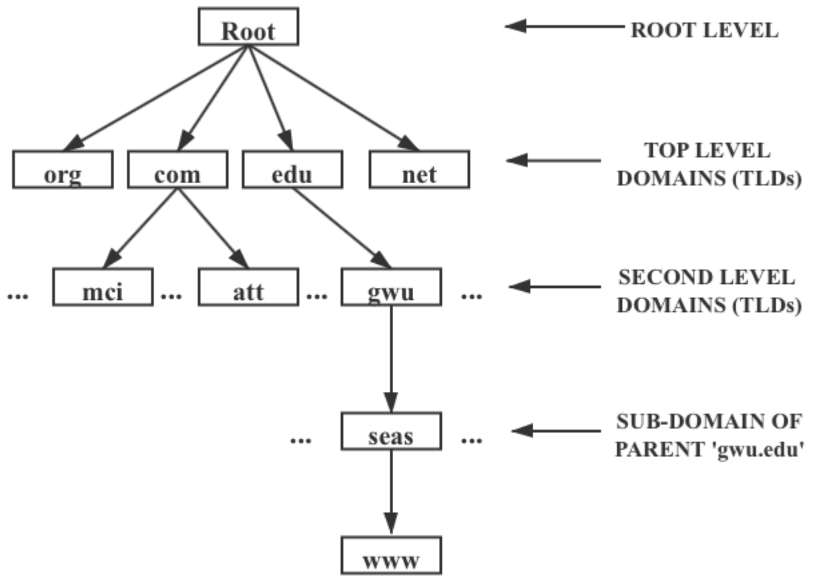


Figure 2. Domain Structure

server if there is no such record;

5) Until you find the correct record, repeat the fourth step;

6) For the next use the local domain name server saves the returned result to the cache and returns the result to the client, too;

The process of domain name resolve is shown in Figure 3.

In order to analyze the security issues of different DNS server levels and protocols, the DNS-related vulnerabilities

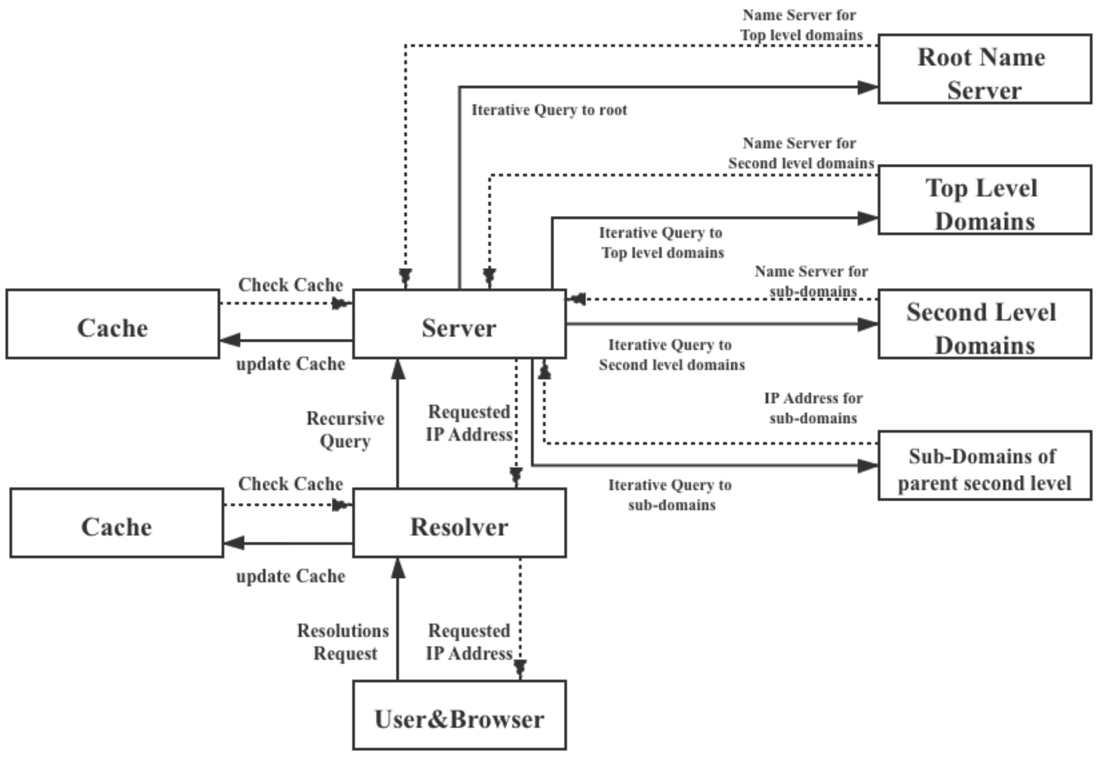


Figure 3. DNS process

are compared and classified here, and the attack targets and attacks are summarized for different types of DNS system intrusions. The statistical results are shown in Table 1.

2.2. DNS Vulnerability Classification

Based on DNS vulnerabilities and related literature , this paper summarizes the vulnerability of DNS as protocol vulnerability, system implementation vulnerability, and architecture vulnerability.

2.2.1. Protocol Vulnerability. The DNS protocol did not fully consider network security issues at the beginning of its design. The UDP protocol is used to transmit information. In the process of message transmission, the data is not encrypted, and the resource records are not signed and protected against forgery. Therefore, the DNS protocol is vulnerable to malicious attacks such as cache poisoning, data eavesdropping, and data tampering.

Poisoning the DNS record buffer is a very common attack method . There are a large number of open parsers on the Internet, which provide attackers with more targets. Related experiments show that 7%-9% of parsers are vulnerable to injection attacks.

The DNS protocol also has the problem of privacy leakage . Since DNS queries contain a lot of sensitive information, in the process of domain name resolution, personal privacy information such as user identity and device type may be disclosed. Many companies use this DNS defect to collect user information, and after analysis and classification, send relevant advertisements to users . By tracking DNS traffic and analyzing user behavior, the accuracy of this method can reach 88% . This shows that attackers can use DNS query traffic to steal user privacy.

There are many attacks using DNS vulnerabilities , such as:

1) Man-in-the-middle attack (MITM): Since DNS does not provide data authenticity and integrity verification, the resolver cannot determine the authenticity and integrity of the received data. Attackers can use this vulnerability to carry out man-in-the-middle attacks, such as packet forgery and transaction ID spoofing;
2) Cache attacks: The use of cache reduces the access time, but it will cause data consistency problems, such as attackers Inject false information into the DNS cache;
3) Distributed Denial of Service (DDoS): If a DDoS attack is performed on the root server, it will have a serious impact on global DNS resolution.

TABLE 1. DNS VULNERABILITY

DNS system vulnerability	Attacking Target	The consequences of the attack
Resource consumption	DNS server	Denial of service
Forged DNS server traffic	Parser	Information leakage Denial of service
Lack of message processing capability	Server, resolver	Denial of service
DNS traffic amplification attack	resolver	Denial of service
DNSSEC key generation protocol vulnerability	DNS protocol	Get the privilege
Buffer overflow caused by abnormal message	resolver, DNS protocol	Get the privilege
Malicious update of DNS records	Name server	Get the privilege
Bypass the access control path	server, resolver	Get the privilege
System configuration error	server, resolver	Information leakage Denial of service
Cache poisoning	server, DNS protocol	hijack Get the privilege

2.2.2. System Implementation Vulnerability. The vulnerability of the DNS system stems from the implementation, composition and management of the DNS system. A large number of open DNS servers on the Internet do not comply with the best configuration principles, nor do they have necessary security verifications. Domain name resolution has serious security risks. DNS service software also has a large number of vulnerabilities. Attacks using zero-day vulnerabilities in DNS service software have become an important means of attacking DNS.

The irregular management of a large number of open third-party DNS servers is one of the reasons for the vulnerability of the DNS system. After years of development of the DNS system, the system has become more and more complex, and the number of servers has increased, including many open DNS servers. In addition to the well-known open DNS servers such as Google DNS and Public DNS, there are still a large number of open DNS servers set up by third-party organizations such as non-governmental organizations and individuals. But these open DNS servers have serious security risks. In the process of domain name resolution, domain name servers and resolvers of different levels and regions may be involved. But in the process of data transmission, no secure link is established, and the authenticity and integrity of the data cannot be guaranteed. The user's personal information and browsing records will be leaked to these open DNS server providers.

The DNS system does not abide by the principle of best configuration, which leads to security risks in the system. At present, the number of open resolvers on the Internet is 10x106 32x106 . The parser uses complex parsing strategies, such as a large shared parsing pool in a tree structure. These resolvers can be divided into three categories:

1) ”entry” servers, which are responsible for receiving user resolution requests;
2) ”egress” servers, which are responsible for interacting with the domain name server and returning analysis results;
3) middleman servers, which are responsible for forwarding Various analysis requests.

Among them, the latter two types of parsers are not visible to end users. However, these open parsers have serious security risks, and only 19% of them can return the correct TTL . In addition, there are a large number of resolvers in the network that users cannot access. These resolvers are responsible for caching the analysis results and improving the analysis efficiency. However, these servers managed by third-party institutions are not standardized enough in terms of deployment and configuration management, and there are security risks .

Vulnerabilities in DNS implementation software are also the main cause of system vulnerability. The Internet Systems Consortium (ISC) investigated the distribution of DNS servers . The top 5 server software used the most are shown in Table 2. It can be found from Table 2 that BIND and Microsoft server software account for more than 95%. Vulnerability information database CVE disclosure , BIND software vulnerabilities up to 102 items, involving BIND 8, BIND 9 and BIND 4.9 and other major versions. Vulnerabilities mainly include DoS, buffer overflow, and permission vulnerabilities. Microsoft's official website announced that the DNS server was attacked by cache sniffing . There is no good way to modify it except for configuration restrictions. Bishop Fox discovered that there is a buffer overflow vulnerability in the Windows DNS client , which may cause malicious DNS responses on Win 8/Server 2012 or later computers. Attackers can use this vulnerability to execute malicious code when the application sends DNS requests. If the attacker controls the DNS server (for example, through a man-in-the-middle attack), he can access the attacked system. In addition, Microsoft DNS servers are also subject to DoS attacks and buffer poisoning attacks .

TABLE 2. DOMAIN SERVER SOFTWARE DISTRIBUTION

Server software	Number	proportion
BIND	85615	80.83
Microsoft	15601	14.73
TinyDNS	2500	2.36
simple DNS	797	0.75
MyDNS	641	0.61

2.2.3. Architecture Vulnerability. The vulnerability of the DNS architecture lies in the single point of failure of the DNS root server. There is also a serious imbalance in the distribution of DNS root servers, and domain name retrieval and control are too centralized.

Although the DNS system adopts a hierarchical structure design, its core is still the root server to manage the entire DNS system. The root server is responsible for maintaining the location information of the top level domain (TLD), and all caches are queried from the root server if there is no hit. There are currently 13 root servers in the world, of which 10 are located in the United States, 2 are located in the United Kingdom and Sweden, and 1 is located in Japan. With the development of the Internet, all countries have expectations of establishing root servers in their own countries, in order to strengthen the participation of the Internet's core infrastructure and enhance the performance of local DNS servers. On June 23, 2015, a global next-generation Internet (IPv6) root server test and operation experimental project based on a new technology architecture-"Snowman Project" was released. In this plan, on the basis of the original 13 root servers, 25 IPv6 root servers will be deployed in 16 countries around the world, in order to realize the multilateral governance of the global Internet. But this only increased the number of root servers, and did not fundamentally solve the vulnerability of the DNS architecture.

The root server is the core of the entire DNS system. If a single point of failure occurs, the entire system will not operate normally and the entire Internet will be severely affected. On October 21, 2002, Eastern Time, 13 root servers were attacked by the most serious and largest DDoS attack in history. Attacks with 30-40 times the amount of data surpassing conventional attacks paralyzed 9 of them. On February 6, 2007, the DNS root server was attacked again by DDoS. The attack lasted for nearly 8 hours, and the source of the attack spread all over the world. In February 2012, the well-known hacker organization Anonymous announced that it would conduct DDoS attacks on 13 root servers around the world, paralyzing the global network by overloading the root servers.

In the domain name retrieval process, the domain name information that the local DNS server does not cache records needs to access the root server to obtain the request address of the domain name server in the next step. In the domain name verification process, the root server of the DNSSEC protocol is deployed as the trust anchor, storing the secret key and signature record of the top-level name server, and providing the final authoritative certification for domain name verification. The root server controls the domain name resolution and verification control to be too centralized, which is the root cause of the fragility of the DNS architecture.

Synthesizing various types of DNS security threats, here summarizes DNS-related threats and specific attack examples.

1) Protocol vulnerability:
    a) Domain spoofing: In January 2014, the top-level domain name in mainland China was redirected to the U.S. IP address 65.49.2.178;
    b) Denial of service: DNS service providers suffered large-scale DDoS attacks in October 2016
2) System implementation vulnerability:
    a) Open server defects: Open DNS server design defects, configuration errors
    b) Software vulnerability:
        i) In September 2016, the BIND vulnerability paralyzed some BIND servers;
        ii) Windows DNS client has a buffer overflow vulnerability and cache poisoning
3) Architectural vulnerability:
    a) Single point of failure:
        i) The DNS root server suffered a large-scale DDoS attack in February 2007;
        ii) The DNS root server suffered a large-scale DDoS attack in November 2015

3. DNS Security Enhancement

To solve the root causes of DNS security threats, this section summarizes and analyzes the enhancements in protocols, systems, detection and monitoring, and architecture in recent years.

3.1. Protocol Security Enhancement

3.1.1. DNSSEC. To solve the problem of the authenticity and integrity protection of the DNS system during data transmission, the IETF (the Internet Engineering Task

Force) proposed the DNS Security Enhancement Act named DNSSEC.

DNSSEC by signing the resource record, the user will also receive the signature of the record when receiving the relevant requested domain name information, and the user can check the authenticity and integrity of the data based on the signature. DNSSEC is based on DNS, adding 4 Security records:

1) DNSKEY records, which store the public key for verifying DNS data;
2) RRSIG records, which store the digital signature of DNS resource records;
3) DS records, which are used for DNSKEY verification, store key labels, encryption algorithms, and the summary information of the corresponding DNSKEY;
4) NSEC record, store the next record adjacent to the corresponding owner, used to deny the existence verification.

Although DNSSEC can solve the problem of false domain name information, it has not achieved good results in its actual deployment process. Although 89% of top-level domains have deployed DNSSEC, the deployment rate of second-level domains. The deployment rate of .com domain names is only 3%, and the deployment rate of .com domain names is only 0.5%. DNSSEC configuration is cumbersome. Each domain uploads its own DS record to the parent domain and signs it by the parent domain, and then uploads it to the root server in turn. The signed record builds a chain of trust: the root domain signs the top-level domain, the top-level domain signs the second-level domain, and then build a chain of trust. However, only about 1% of .com, .net, and .org domains have deployed DNSSEC. Among the domain names deployed with DNSSEC, more than 30% of the domain names caused configuration errors due to a lack of DS records.

Among the clients that have deployed DNSSEC, many of them still have resolution failures. A research institution has conducted a large-scale measurement of more than 500,000 clients using DNSSEC name resolution. The measurement results found that only a small number of users are affected by DNSSEC. Verification protection, DNSSEC deployment will lead to end-to-end resolution failures. On average, about 1 of every 10 clients deploying DNSSEC cannot access the domain name. In addition, DNSSEC is difficult to solve the interoperability and information reference problems between different domains.

DNSSEC uses a public key cryptosystem, which also introduces additional overhead during the encryption process. Due to the overhead of DNSSEC message encryption, a large number of DNS requests cause the server to frequently calculate signatures, which increases the server response time, and a large number of messages containing digital signatures will also Occupy bandwidth resources.

3.1.2. DNSCurve. To provides link-layer security protection for the DNS system, DNSCurve encrypts all DNS data packets during transmission through an elliptic curve encryption algorithm and key distribution mechanism which will make sure its confidentiality and integrity protection.

The working process of DNSCurve is as follows:

1) DNSCurve server first embeds the public key into the NS resource record through the encoding mechanism. The client looks up the NS record, combines with the predetermined encoding mechanism, and finds the corresponding public key;
2) The client adds the public key of DNSCurve to a The nonce is put together in the cryptographic box, and these are encoded as an extended DNSCurve query packet;
3) The DNSCurve name server will first verify the client's request packet, if it is correct Data, the response data packet is encrypted by the lockbox and sent to the client as an extended response data packet; otherwise, the DNSCurve server sends to the client according to the error reason of the request data packet, and the client receives the response data packet, Re-modify or discard.

Although DNSCurve provides confidentiality and integrity protection during transmission based on the DNS protocol, the recursive resolver that uses DNSCurve has no way to inform the name server about the validity of the response, so the name server can only blindly trust the Local recursive resolver. In addition, DNSCurve needs to modify the DNS protocol, which limits the incremental deployment of DNSCurve.

3.2. System Implementation Enhancement

3.2.1. Transmission Protocol. The current DNS protocol uses the UDP protocol to transmit data, and the authenticity and integrity of the information are not verified. Therefore, enhancing the DNS transmission protocol is a means to improve DNS security. T-DNS uses TCP and TLS protocols instead of UDP to transmit DNS messages. The resolver and the server first need to establish a TCP connection, and then use the TLS protocol to encrypt and protect the content of the DNS message to prevent content leakage and malicious tampering. T-DNS uses the number limit mechanism of TCP connections to prevent malicious servers from actively pushing false response information. At the same time, the TLS protocol is used to protect the security of data transmission and solve the problem of data leakage and malicious tampering. The limitation of this method is that the time overhead of establishing a TCP connection will affect the resolution efficiency. T-DNS uses the TCP and TLS protocols, which are compared with the traditional DNS Incompatible and difficult to deploy on a large scale.

3.2.2. Query Mechanism. Multiple DNS servers work together to request other servers when a single server fails to resolve, which improves the robustness and availability of the system. Negotiation of the query results of multiple DNS servers will also improve the authenticity of domain name resolution.

Multiple DNS servers work together to complete the domain name resolution process, which is a way to improve the robustness and reliability of the system. The CoDNS system uses local and proximity-aware design ideas to distribute DNS requests, and achieve low-latency and low-overhead names Resolution. When the local DNS server fails (packet loss, overload, or configuration error), CoDNS automatically redirects the request to a healthy collaborative server, which effectively reduces latency and improves the reliability of query services. Although collaborative DNS improves Reliability and performance, the security of the system will also be reduced, and single points of failure and failure will easily spread to the entire system. In order to solve this problem, the ConfiDNS system uses multi-site agreement (multi-site agreement) and each site query History to solve the shortcomings of the cooperative inquiry process.

Compare the query results of multiple DNS server software to reduce the impact of DNS software defects or vulnerabilities on the accuracy of the query results. The DR-DNS system can run multiple copies of different DNS server software at the same time. The user first requests DNS Send to DR-DNS, DR-DNS distributes the request to different DNS software copies, and various DNS server software returns the query results to DR-DNS. DR-DNS uses a voting mechanism to select the record with the most occurrences and return it to the customer. The advantage of this design is that it can still respond correctly to the client under the parsing error caused by a certain DNS software vulnerability attack and software defect. However, the resolution efficiency of DR-DNS is low due to the processing overhead of the voting mechanism and the response speed of different DNS servers.

3.2.3. Information Confidentiality. The current DNS protocol does not encrypt and sign the transmitted information, the authenticity and privacy of the domain name query results cannot be guaranteed, and there is a risk of tampering with the query results and privacy leakage. On the basis of the existing DNS system, Change the organizational structure and transmission protocol of the DNS system to improve the authenticity of the resolution results, and protect user privacy.

In order to prevent malicious attacks on the top-level domain name server, OnionDNS hides the top-level domain name (TLD) and introduces a mirror server and .o domain as the top-level domain of OnionDNS. The mirror server is updated synchronously with the .o domain managed by the root server through the Tor network. Mirror The server is located between the client and the root server. The client first sends the request to the mirror server through the open network or Tor network, and then the mirror server first looks for the cached .o domain record, and if there is a cache record, it returns the query result. If the cache record expires or there is no cache record, the mirror server will not initiate a query request to the hidden root server to prevent attackers from injecting traffic into the Tor network and affecting the anonymization service. The mirror server periodically obtains it from the hidden root server The update record of the corresponding .o domain, and the result is returned to the client. Since the mirror server uses the DNSSEC protocol, authenticity can be guaranteed when communicating with the hidden root server. In this way, the root of anonymity can be achieved. The purpose of the server to provide security protection. In order to prevent malicious traffic from being injected, the mirror server does not query the root server when the cache misses, instead of updating the resource record, which leads to a reduction in resolution efficiency.

To solve the privacy leakage problem caused by third-party DNS resolvers, an additional server is introduced into the traditional DNS resolver and DNS server to achieve the effect of privacy protection. As an alternative to traditional DNS, EncDNS is based on the DNSCurve protocol. Encapsulate the encrypted query content in a standard DNS message and add an EncDNS server between the resolver (conventional resolver, CR) and the DNS server. CR sends a resolution request to the EncDNS server, and the EncDNS server obtains the decrypted domain name information and sends it to the authorized server Initiate a query, and then encrypt and encapsulate the query result, and then return it to the client through CR. Since the query information initiated by the client is encrypted, the query content is encrypted. CR does not know the query content, after CR forwards the query content to the EncDNS server, the EncDNS server The query content is decrypted, but the IP address of this query request is the address information of CR. EncDNS does not know the specific user IP address. Therefore, EncDNS and CR complete the resolution process together and do not know the complete request content of the user, achieving privacy The purpose of protection. The limitation of EncDNS is that it needs to modify the DNS protocol to transmit encrypted query information, which is not conducive to widespread deployment.

3.3. Detection and Monitoring

With the development of the Internet, technology is constantly advancing, and attack methods are constantly changing. Only relying on protocol enhancements and system changes may not be able to resist all attacks. Therefore, on the basis of existing systems, effective monitoring and diagnosis to protect the normal operation of the DNS system is also an important security enhancement. The diagnosis and monitoring of the DNS system do not need to change the existing DNS implementation methods, and it has good gradual deployment capabilities and can effectively monitor various attacks. Moreover, the core idea of detection and monitoring is to analyze and detect DNS query traffic, construct a detection system, and use technologies such as machine learning and information entropy to learn and classify the detection results to improve detection accuracy. This section is divided into monitoring DNS according to the level of detection traffic between the client and the recursive server and the traffic between the DNS servers are detected.

3.3.1. Monitoring the Traffic Between the DNS Client and the Recursive Server. BotGAD detects botnets by analyzing the traffic between bots and control centers, DNS servers, and other hosts. BotGAD maps group activities to vectors, calculates the similarity between the vectors, and determines whether members of the group activities have participated in malicious activities. The mapping method is to generate a group of activities in each time interval and introduce a binary relationship table. The horizontal rows of the table represent each time interval, and the columns of the table represent the IP addresses of group members. If within a certain time interval, If the IP of a certain group member participates in the activity, the data corresponding to this time interval and IP address is 1, otherwise it is 0, extract the information of each column of the table, and this column vector represents each IP address in each time interval Participate in group activities, and then calculate the average similarity between vectors. If the threshold is exceeded, it indicates that the group activities have participated in the group activities of the botnet.

Seguio uses the communication traffic between the client and the local DNS resolver of the ISP network to construct a machine-domain name bipartite graph model. Each node in the graph represents a host and a domain, and the edge indicates that the host queried the domain name during the observation phase. Each domain node is attached with a comment, including IP address, domain activity (such as the time of the first query). Nodes that are connected to known malicious nodes are marked as malicious nodes and will belong to the domain name whitelist (such as according to alexa. com) is marked as a benign domain. For unknown nodes that cannot be marked, they are classified by the characteristics of ”who queried whom”. If a malicious host queries an unknown domain, or an unknown host queries a known malicious Domain, the node has a high probability of being a malicious node.

Moreover, monitoring lower-level DNS traffic can also be used to detect botnets and DNS tunnels that use DNS traffic control. The clustering algorithm is used to classify DNS traffic characteristics, and the nodes in the botnet are similar to each other in the same category. There are obvious differences between the nodes in the botnet, and the existence of botnets can be detected in this way. Using the time and space distribution relationship in the DNS query traffic combined with the information entropy feature is also a means of detecting DNS traffic control botnets. Detecting the DNS tunnel currently mainly uses detection technologies such as DNS traffic feature matching and statistical analysis of traffic analysis.

3.3.2. Monitoring Traffic Between DNS Servers. Based on the historical information of DNS collected from multiple recursive servers, the Notos system uses network characteristics, regional characteristics, and blacklist evidence characteristics to construct a model for the legal allocation and operation of network resources and uses these models to calculate reputation scores for new domain names. If the reputation score is too low, it means that this new domain name is involved in malicious activities.

Compared with Notos, EXPOSURE is not based on historical DNS traffic, but detects real-time real traffic information, and defines 15 malicious domain behavior characteristics, and divides these characteristics into time-based, DNS response, TTL value, and domain name information4 These features are stored in the feature classification module. The data collection module collects DNS data, and the feature classification module classifies these data according to 15 pre-set malicious features. Also, the domain name collection module and the data collection module run at the same time, and then use the domain name. The known illegal and legal domain names were collected by the acquisition module mark the data classified by the classification module. For domain names that can be judged as legal or illegal, they are used as the input of the training module for training, and the classifier module is judged according to the detection model of the training module. Whether the unmarked data is legal or illegal. Among them, the classifier module uses the J48 decision tree algorithm, and uses the concept of information entropy to construct a decision tree using marked data. The limitation of the DNS traffic detection scheme on the local recursive server is It is necessary to set up a large number of data collectors on the recursive server to obtain global information of a domain. However, it is difficult to be applied in practice due to the privacy of data and the deployment of collectors.

To detect malicious domain names, high-level DNS traffic was monitored to get a global view of the DNS domain. Research results based on this idea are Kopis. The obvious difference between Kopis and Notos and EXPOSURE is the level of monitoring. Notos and EXPOSURE monitor Recursive server, and Kopis monitors top-level name servers and authorized name servers, and monitors DNS data by dividing different timings. Kopis has two operating modes: training mode and operating mode. In training mode, the training module will be known to be legal With the DNS data of the illegal domain name set KB within m days as input, the feature vector set Vtrain is generated. The domain name in each KB corresponds to a label in Vtrain that is legal or illegal, and then supervised learning technology is used to construct a DNS legal and illegal query feature. Statistical classification model S, these query data come from the upper layer traffic of DNS. In the operation mode, the classification model S is applied to set a credible score for unknown domain names in a certain time sequence. To give the final score, the model S will calculate whether the average credit score in m consecutive time series is lower than the preset threshold and the domain name is judged to be illegal. The advantage of Kopis is that it can be configured independently and does not need to share data from other networks, so it does not need to consider the security problems from different organizations and regions and privacy issues of information sharing. However, the limitation of Kopis is that for domain names with a short survival time, such as domain names generated by the DGA algorithm, the detection efficiency is very low.

As for detecting malicious activities such as botnets, phishing websites, and spam in the ISP backbone network that threaten user safety, DAOS detects DNS traffic flowing through the backbone network boundary in real-time, and characterizes DNS activities from two aspects: domain name dependence and location of use Among them, the dependency observes the external use of the domain name from the user's perspective, while the location focuses on the internal resource configuration of the domain name recorded in the zone file, and uses supervised machine learning algorithms to detect them, and then combines the two aspects through a weighted average. Detection results, timely and accurate identification of malicious domain names.

3.4. Architecture Enhancement

As the core of the DNS system, the DNS root server is responsible for the maintenance and management of the DNS home directory. There are flaws in this system, such as a single failure point and weakness. Some researchers have suggested developing a modular DNS structure to address the issue of DNS centralization. After the DNS system is decentralized, each server node is equal, and the impact of single points of failure and DoS attacks will be reduced. The DNS resolution process is no longer limited to the root server, and domain names will not be blocked due to management and other factors, which also solves the problem of the limited number of root server deployments.

3.4.1. Fully Distributed System Structure. With the emergence of the P2P network structure, taking advantage of the fault tolerance and load balancing characteristics of the P2P network, some scholars have proposed a domain name resolution service based on the P2P network. The domain name system of the P2P structure is adopted, and the nodes are equal to each other, and traditional DNS servers will not occur. The centralization problem of a single point of failure.

One solution to establish a fully distributed DNS is to use distributed hash table technology to construct a decentralized network. Then, use the characteristics of distributed hash table fault tolerance and load balancing to achieve improvements to the existing DNS system, such as Chord-based DDNS, P-DONAS based on Kademlia. DDNS searches for resource records on traditional DNS servers, and the retrieval and storage of DNS records are done through distributed hash tables. DDNS is characterized by inheriting the fault tolerance and load balancing of DHash, etc. Features. In terms of load balancing, consistent hashing is used to evenly distribute secret keys to each stage. The question route is cached while each node is retrieved. This query method's time complexity is O (logN). In terms of robustness, as servers join and exit, distributed hash tables automatically transfer data, so these data will always be stored on a fixed number of servers. Since these servers are selected in a pseudo-random manner, data will be lost only by all servers paralyzed at the same time. P-DONAS uses each domain name provider (ISP) site as an access node AN (access node). These ANs are responsible for interacting with the client and are also responsible for storing resource records. The AN which usually accessed by the user is called the triggering node. When receiving a user request, the triggering node first looks for its own cache record. If it is not found, it searches on the P2P network. The traditional DNS service will be queried. If the result is found, the result will be returned to AN. AN will return the result to the client and cache it in the buffer. If the result is still not found on the traditional DNS server, the record will be returned. Exist or end the query over time. When there is no cache record in the P-DONAS system, the traditional DNS server will be searched. Therefore, P-DONAS is also compatible with the traditional DNS server.

To improve the retrieval efficiency based on the distributed hash table domain name system, the Beehive active caching mechanism can achieve an average search time complexity of O(1) and support fast updates. The CoDoNS system adopts this design idea and can be compared with traditional The DNS system is compatible to achieve a smooth transition. CoDoNS is composed of distributed nodes around the world. These nodes self-organize to form a P2P network. The domain name records are cached through the home node. If the home node fails, its neighboring nodes will become a home node. CoDoNS uses the same protocol and transmission format as traditional DNS, and the client resolver does not need to be modified. CoDoNS separates the management of the namespace from the traditional DNS, and the domain name owner only needs to purchase the name certificate from the domain name provider. The domain name provider can add them to CoDoNS, and the domain name owner does not need to provide a dedicated server for the domain name. The query resolution process of CoDoNS is very simple. The client sends a DNS query request to CoDoNS, and CoDoNS obtains records at home nodes or at intermediate nodes Obtain cache records and send response information to the client. In addition, the home node will interact with the traditional DNS server to keep the stored records up to date and update the cache information.

The P2P-based DNS system is easily affected by the network environment, and the query efficiency will be reduced when the network fluctuates. In order to solve the structural problems of traditional DNS and the efficiency problems based on P2P networks, HDNS combines P2P with traditional DNS systems and proposes a hybrid structure DNS System solution. The system is divided into two parts: public zone, where nodes are organized by P2P network; internal zone, where nodes are organized in a tree structure of traditional DNS. All nodes in the public zone are organized by A unique identifier is assigned, and the root node in the tree structure of the internal area is also assigned a unique identifier, and the root node identifier is mapped to the identifier of the common area. In this way, each internal area and the common area perform Association. In view of efficiency and security performance, the top-level domain name and second-level domain name are stored in the public area, and the rest is stored in the internal area. When querying, first query the identifiers of the top-level domain name and the second-level domain name in the common area, get the root of the internal area through mapping, and the identification of the node. Then look up the remaining part of the record under the root node, and finally return the query result. Due to the hybrid structure of HDNS, the security is higher than that of traditional DNS, and the query rate is faster than the domain name system based entirely on the P2P network. Although the domain name system based on the P2P network has the advantages of robustness and load balancing, the domain name system based on the P2P network also has the following limitations.

1) The worst-case query delay is unacceptable: P2P networks have different processing delays due to different underlying implementations, but the worst-case query delay is unacceptable, such as a query request may be in multiple high-latency networks It is processed after multiple transmissions, and the analysis efficiency is significantly reduced.
2) Updating node information results in an unstable state: the P2P network enables data to be changed from either node. When a node changes the data without broadcasting, it will be disconnected, which will lead to an inconsistent state of network nodes. Some nodes store expired domain name information.
3) Data forgery: P2P networks do not have data write rate limits and access control mechanisms. In order to expand to the whole network, attackers will flood the entire P2P network with a huge volume of junk data and fake some fake domain name information.

3.4.2. Domain Name Structure Based on Blockchain.

Blockchain is a modern distributed data storage technology mode, point-to-point transmission, consensus process, algorithm of encryption and other innovations.Using the characteristics of blockchain decentralization, non-tampering, traceability, high credibility and multi-party maintenance, decentralization for design DNS provides new ideas.

Namecoin is a DNS system developed based on Bitcoin. Based on the Bitcoin system, the transaction information stored in the blockchain is replaced with name-value mapping data. Therefore, Namecoin and Bitcoin have most of the common functions and mechanisms, but Namecoin is one A more general name-value pair resolution system, rather than a replacement for the current DNS system. Namecoin matches other types of names by using different prefixes-value pairs. For example, the ”d/” prefix is used in domain names, ”id/ ”The prefix is used to register the identity. Namecoin uses the top-level .bit virtual domain name, but this domain name has not formally been licensed with the new DNS system. Namecoin and DNS systems are isolated. If additional analysis software is not installed, the DNS system cannot Resolve domain names in bit.

The bottom layer of Namecoin is implemented by the Bitcoin system. It only replaces stored information from transaction data with name-value mapping information, which has limitations in expansion. In order to solve the scalability problem of Namecoin, Blockstack proposed a solution to layer domain data and control. The domain name records are stored in an external database, and the underlying control is implemented by the blockchain. Blockstack only stores a small amount of metadata in the blockchain (i.e. data hashes and state transitions) and uses external storage to store actual large blocks of data. The registration protocol, readable name data, development of name hash binding, and secret key binding are specified by the control plane. The data plane is responsible for data management and confirmation of availability. The Blockstack is composed of 4 layers: the control plane includes the blockchain layer and virtual, chain layer, the data plane contains the routing layer, and the data storage layer. The first layer of the blockchain layer is responsible for storing the sequence of blockchain operations and providing a consensus on the writing order of the operations. The second layer is the virtualization layer, which defines New operations, but do not need to change the underlying blockchain layer, only Blockstack nodes know these operations, and the underlying blockchain nodes do not. Moreover, in the virtualization layer, the approval and rejection rules of Blockstack operations are also specified. The third layer is the routing layer. Blockstack separates the routing request from the actual stored data. This avoids the system from adopting any specific storage service from the beginning. Instead it provides for the coexistence of various computing services, including private cloud storage and P2P. Just like traditional DNS uses zone files, Blockstack also uses zone files to store routing information. The fourth layer of storage is at the top, and it stores the actual data of name-value pairs. The data is separated by the hidden key signature of the owner and users do not need to trust the storage layer so the accuracy of data values in the control plane can be checked. The storage layer has two storage methods: mutable storage and immutable storage. The difference between the two methods is the data integrity verification and Blockstack supports these two methods to run at the same time. In comparison, there are also a host of blockchain technology-based Bitcoin-derived applications, which still offer name resolution facilities, such as ethereum-based ens, peername, Emercoin EMCDNS. There are still the following drawbacks to the blockchain-based domain name framework.

1) Client browsers must mount plug-ins that are incompatible with existing DNS systems to access the domain name system, so it is difficult to deploy a blockchain-based domain name system on a large scale.

2) The underlying implementation of Namecoin and Blockstack are both Bitcoin. Because Bitcoin adopts the "one-CPU-one-vote" mechanism. If an organization controls 51% of the entire system's computing power, that is, a 51% attack, it will cause serious damage to the system Security risks. Although 51% of attacks are theoretical, they can threaten the security of the system if they have 25% of the computing power.

3) As the blockchain stores all historical information, the entire system will become larger and larger, and it is difficult for mobile devices or personal computers to have enough hard disk space to store all recorded information. Although some scholars have proposed the SNV (simple name verification) protocol, But you need to set up a server that provides all records, and the communication security between the server and the client is another problem that needs to be solved.

3.4.3. System Structure Based on Root Server Alliance. The centralized resolution of the DNS system contains the risk of abuse of rights, that is, a top-level domain may be deleted, causing the subdomains under the entire top-level domain to be unresolvable. In order to solve the problem of centralization of the DNS root server, improvements should be made from both the structure and the resolution mechanism.

In terms of the root server structure, the following technologies are mainly used to decentralize the root server:

1) Recursive root: Root zone resolution is directly performed on the recursive server;
2) Root camouflage: The root zone query is directed to the mirror root service analysis ;
3) Open Root: establish a set of independently operating root servers, using IANA's root zone data as the analytical data source;
4) Global Root: by increasing the number of root servers, using anycast technology, expand 13 root servers to a larger scale, the root zone data in these four schemes still come from IANA, and the analysis of rights abuse still exists.

In terms of the resolution mechanism, the domain name peer-to-peer diffusion method is adopted, that is, the owners of each top-level domain name report the address of the top-level server to the masters of other national top-level domain names. Under the domain name peer-to-peer diffusion system, the autonomous root and the international root server are in a mixed work State, the autonomous root transfers the current centralization of the root server to the top-level domain name server, but if the top-level domain name server such as .com refuses to transfer authoritative information to the autonomous root, the autonomous root will be greatly restricted.

The root server rights are weakened into multiple sub-nodes, and the DDNS system uses this method to limit the root server rights. DDNS is based on the Paxos distributed consensus algorithm and manages domain names hierarchically and regionally. Just more than half of the root nodes will transfer the root zone transaction order. To keep it from relying on the primary root server. The essence of this solution is to delegate the authority of the root server to each sub-root, and determine the transaction request through the voting mechanism, but the limitation of this design is that the sub-root servers will align with each other and vote for the results Under the control of the alliance, which affects normal transaction requests.

3.5. Summary

The security situation of DNS is severe. In order to deal with various DNS security threats, this section summarizes the current DNS security solutions from four aspects: protocol enhancement, system enhancement, detection and monitoring, and decentralized domain name system. Moreover, a comparative analysis was carried out to analyze and compare the advantages and limitations of various enhancement schemes more intuitively. Table 3 summarizes the aspects of client compatibility, protocol backward compatibility, privacy, delay, the resistance to DoS attacks, and the resistance to cache poisoning attacks.

4. Common DNS Sttacks and Precaution

To keep the security of data, large companies are now spending more attention to DNS defense because DNS attacks can be realized in many ways. These attacks usually contain Domain hijacking, DNS Cache poisoning, DDOS attack, DNS spoofing, and Reflective DNS amplification attack. In this section, the implementation of these attacks and preventive measures will be discussed.

TABLE 3. Comparison of various DNS enhancement schemes

Enhanced Program	Content	Comparison					
		Protocol compatible	Compatible with traditional DNS	Privacy	Anti-DoS attack	Anti-cache Poisoning	Low latency
Protocol enhancement	DNSSEC	yes	yes	no	no	yes	no
	DNSCurve	yes	yes	yes	Part	yes	no
System enhancement	T_DNS	no	no	yes	no	yes	no
	EncDNS	no	yes	yes	no	yes	no
	CoDNS	yes	yes	no	yes	Part	yes
	CofiDNS	yes	yes	no	no	Part	yes
	DR-DNS	yes	yes	no	no	no	no
P2P structure	DDNS	no	no	no	yes	yes	no
	CoDoDNS	yes	yes	no	yes	yes	yes
	P-DONAS	yes	yes	no	yes	yes	no
Blockchain structure	Nameecoin	yes	no	yes	yes	yes	no
	BlockStack	yes	no	yes	yes	yes	no
	ENS	yes	no	yes	yes	yes	no
	Peername	yes	no	yes	yes	yes	no
	EMCDNS	yes	no	yes	yes	yes	no

4.1. Domain hijacking

Domain hijacking is usually the responsibility of the DNS service provider and users can do nothing about it. Attackers use hacker techniques to access the login for domain name management and the mailbox for domain name management. Attackers then point the domain name NS record to a DNS server that can be managed by the hacker and link the corresponding domain name record to the DNS server. Therefore they would enter the content referred to by the hacker as users browse the domain name.

The concept of domain hijacking is shown in Fig.4.

4.1.1. Domain hijacking attack classification. DNS hijacking may be categorized according to the client side-recursive DNS domain-authoritative DNS server route as follows:

1) Local DNS hijacking:
   DNS hijacking on the client side is collectively referred to as local DNS hijacking. Local DNS hijacking may be:

   a) To enter the PC and tamper with the DNS settings, hackers use Trojan horse viruses or malware to (hosts file, DNS server address, DNS cache, etc.).

   b) Hackers use router vulnerabilities or break the router management account to invade the router and tamper with the DNS configuration.

   c) For internal business situations, certain enterprise proxy devices (such as Cisco Umbrella intelligent proxy) conduct DNS hijacking and resolution of unique domain names.

2) DNS resolution path hijacking:
   DNS hijacking that happens during the DNS resolution process during network contact between the client and the DNS server is universally known as hijacking of DNS resolution paths. DNS resolution route hijacking can be divided into the following three groups by splitting the hijacking path of DNS resolution messages at the query stage:

   a) DNS request forwarding: Redirecting DNS traffic by technological means to other DNS servers (middle box, software, etc.).

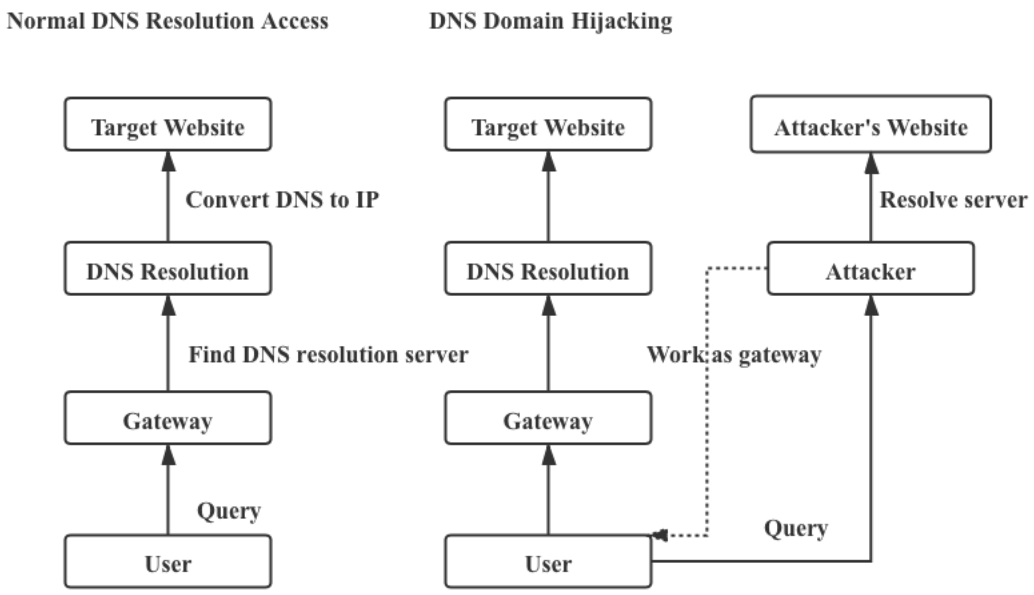


Figure 4. DNS Domain Hijacking

b) Replication of DNS requests: The DNS request is copied to the network computer using devices such as spectroscopy, and the output of the DNS hijacking is returned prior to the usual response.
Case: A DNS query packet capture returns two different responses.

c) Pickup DNS request: Network equipment or program replaces the DNS server directly in order to respond to DNS queries.
Case: Some DNS servers implement the functions of SERVFAIL rewriting and NX-DOMAIN rewriting.

3) Tamper with DNS authoritative records:
Tampering with DNS authoritative records To explicitly alter DNS records, we apply here to the actions of hackers secretly breaking into the DNS authoritative record management account.
Case: Hackers hacked into the management account of the domain name and tampered with the DNS authoritative record to point to their own malicious server to achieve DNS hijacking.
The attack is shown in Figure 5.
Case: The hacker broke into the upper-level domain name registry's management account, tampered with the domain name's NS authorization log, and allowed the domain name to be stolen by a malicious DNS server created by the hacker himself.
The attack is shown in Figure 6.

4.1.2. Domain hijacking defence. There are many methods to prevent Domain hijacking:

1) Set complex passwords for domain name registrars and registration mailboxes and change them frequently. Also, users should be careful to use the same username and password in multiple important registration locations.

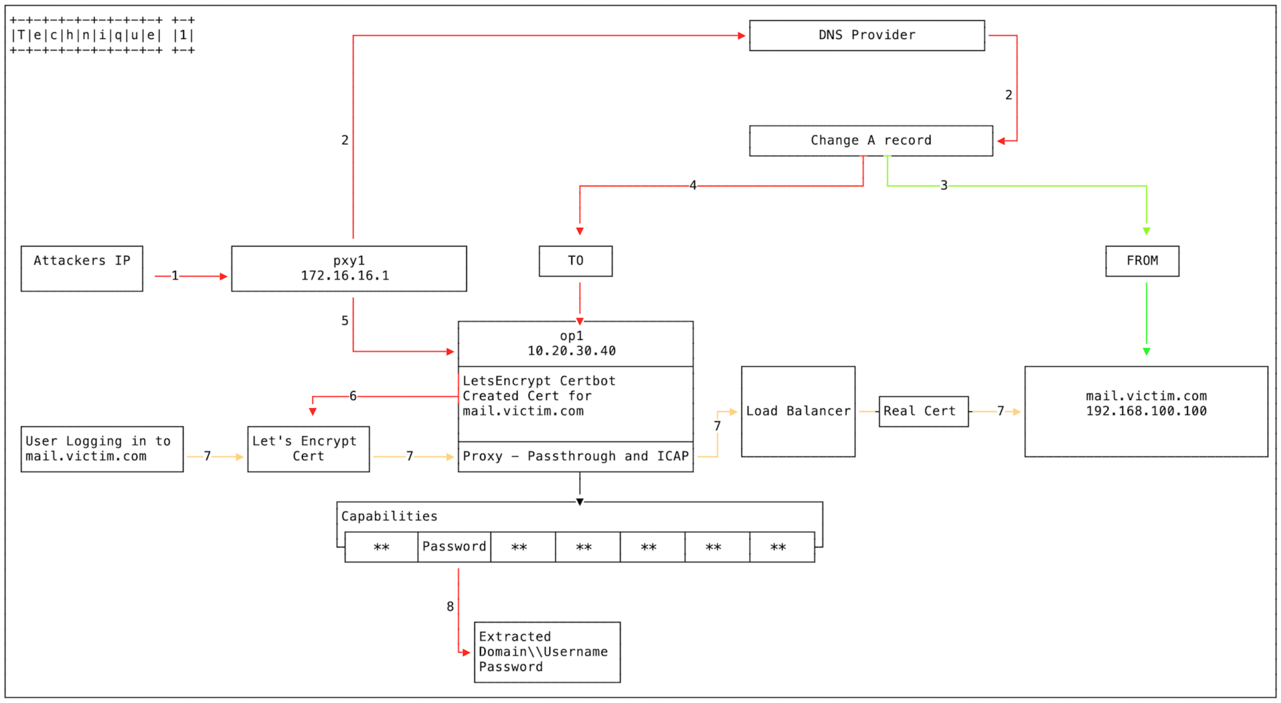
2) Set the domain name update to a locked state, and the website of the DNS service provider does not allow you to change the record. The domain name resolution that needs to be performed must be achieved by the service provider by using this process.

3) Check the domain name account information, domain name whois information, and the event manager regularly. Moreover, it is important to clean up suspicious files, website index and external link information on the Website.

4) Strengthen the website's anti-SQL injection functionality. SQL injection is a mechanism by which SQL statement functions are used to write material to the database to gain permissions.

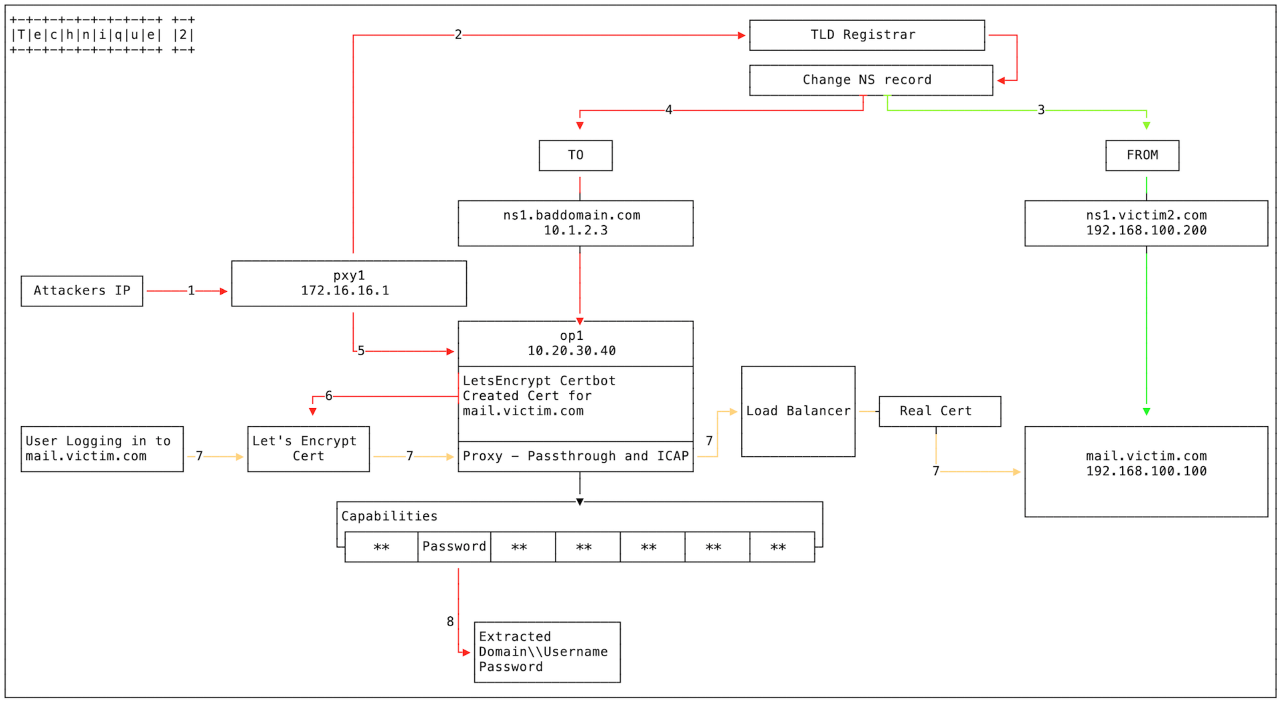
5) Disable redundant utilities operating on the DNS registry, such as FTP, and customize the website folder and file operation permissions.

6) Use firewall services on the network perimeter and



|T|e|c|h|n|i|q|u|e| |1|
DNS Provider
2
Change A record
4
3
Attackers IP
1
pxy1 172.16.16.1
TO
FROM
5
op1 10.20.30.40
LetsEncrypt Certbot Created Cert for mail.victim.com
6
Load Balancer
Real Cert
7
mail.victim.com 192.168.100.100
User Logging in to mail.victim.com
Let's Encrypt Cert
Proxy – Passthrough and ICAP
Capabilities
** Password ** ** ** ** **
8
Extracted Domain\\Username Password

Figure 5. DNS A Record



|T|e|c|h|n|i|q|u|e| |2|
TLD Registrar
2
Change NS record
4
3
TO
FROM
ns1.baddomain.com 10.1.2.3
ns1.victim2.com 192.168.100.200
Attackers IP
1
pxy1 172.16.16.1
5
op1 10.20.30.40
LetsEncrypt Certbot Created Cert for mail.victim.com
6
Load Balancer
Real Cert
7
mail.victim.com 192.168.100.100
User Logging in to mail.victim.com
Let's Encrypt Cert
Proxy – Passthrough and ICAP
Capabilities
** Password ** ** ** ** **
8
Extracted Domain\\Username Password

Figure 6. DNS NS Record

DNS server. Limit access to those ports/services required by DNS functions.

4.2. DNS Cache Poisoning

DNS cache poisoning, also known as DNS cache pollution, refers to some deliberately manufactured (poisoned) or unintentionally manufactured (polluted) DNS packets that cause the DNS cache server to cache errors Domain name resolution record.

DNS cache poisoning works: because there is no authentication method for the regular DNS query, and the UDP on which the DNS query is normally based is a connectionless and insecure protocol, it is very straightforward to tamper with the DNS query. The database is tracked by inspecting the DNS on UDP port 53, and if a request matching the keyword is detected, it automatically pretends to be the target domain name's resolution server (NS, Name Server) and returns an error response to the query.

In order to reduce the traffic on the network, the general DNS cache server will cache the domain name data, and the service can be provided immediately when another DNS client requests to resolve the same domain name next time. When the cache server caches the wrong domain name data, the DNS client will get the wrong result when requesting these domain names.

Attackers use the DNS cache server access to take the user who was browsing a certain unknown website to other websites that the hacker referred to.

DNS actually uses the UDP protocol for transmitting question and response data packets, and adopts a basic mechanism of trust. For the first received DNS response data packet, only the original query packet sending IP address, port and ID are confirmed, and the validity of the data packet is not affected. If it fits, do some review, accept it as the right response packet, resume the DNS resolution process, and discard all subsequent arriving response packets. This makes it possible for an attacker to mimic an authoritative DNS server and send a falsified response packet to a DNS cache server and first try to complete a response to contaminate the DNS cache. If the forged response packet sent by the attacker arrives at the cache DNS server prior to the correct response packet sent by the authoritative name server and matches the original query packet's IP address, port, and ID, the DNS cache may be effectively poisoned.

Cache poisoning can be accomplished in many ways. For eg, it can be attacked or managed by manipulating the loopholes on the ISP side of netizens in the DNS cache server, thus altering the response results of users accessing domain names on the ISP. In addition, some hackers are able to find weaknesses in the authoritative domain name registry of the customer. This means that hackers will enforce cache poisoning and store the incorrect domain name record in the cache as the authoritative domain name server of the user can be used as a cache server at the same time. The cache then helps to achieve the incorrect DNS resolution outcomes for the users used. This is the way the major defects of DNS have been discovered recently and It is only said to be a "major" defect. It is reported that the architecture and execution of the protocol itself is causing it. There are such issues with nearly all DNS applications.

Let's take a look at several types of poisoning:

1) The "birthday attack" of poisoning caused by random response packets: Before 2008, all DNS request packets used fixed source port 53 to send resolution requests. Therefore, apart from the ID, all the information needed to spoof the DNS reply is limited. Attacking DNS with this weakness is called the "birthday paradox", and it takes an average of 256 times to guess the ID. In order for the attack to be successful, the fake DNS reply must reach the caching DNS server before the legitimate authoritative DNS server responds. If the bogus answer comes first the caching DNS will cache it and before its time to live (TTL) expires, the authoritative DNS will not be required to address the same domain name by recursive DNS.

2) Kaminsky cache poisoning: In 2008, someone announced a new type of cache poisoning principle on Black Hat. The basic random guessing technique remains unchanged. This attack uses the AUTHORITY SECTION field of the DNS response packet, because the DNS response can be a direct response (direct IP address of the request) or a reference (a server authoritative for a given zone). The basic idea is that the attacker chooses the domain they want to attack, and then queries the target resolver for subdomains that have not been cached by the resolver (querying non-existent subdomains is a good choice, because non-existent subdomain records are not DNS server cache). Although the subdomain is not in the cache, a query is sent to the authoritative registry for that domain by the DNS recursive server. At this point, the attacker floods the usual response with a huge number of forged responses, and there is a different forged ID number in each forged response. If the attacker successfully injects a forged response, the recursive DNS server will cache the wrong IP for the authoritative server. Future DNS queries to the compromised domain's recursive DNS server would trigger all requests to be redirected to the authoritative resolver of the attacker controller, allowing the attacker to send malicious answers to any new DNS record without inserting fake entries.

A example of DNS cache poisoning is shown in Figure 7.

Users should aim to pick DNS applications or a variant with strong source port randomness to avoid DNS Cache poisoning. As the party responsible for the authoritative domain name, the deployment of as many authoritative DNS as possible, such as one master and many backups, is important. New protection mechanisms focused on current DNS, such as the deployment of DNSSEC or the addition of new security authentication protocols, should also be binding.

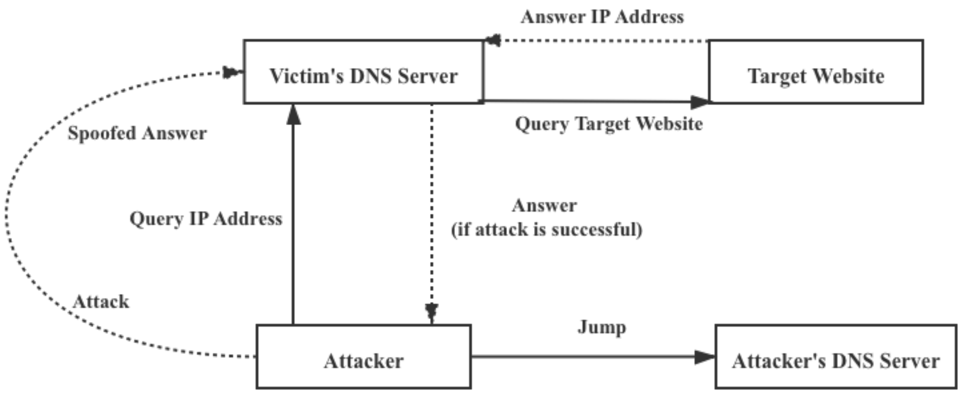


Figure 7. DNS Cache poisoning

Implementing DNSSEC, a reliable form of encryption and authentication, is the safest way to avoid DNS cache poisoning. As the backbone and cornerstone protocol of the entire Internet, DNS is currently the most commonly used implementation method for unencrypted plaintext transmission. DNSSEC is a better method for DNS authentication. The DNS server's response to the DNS resolver uses the DNSSEC signature method. The DNS resolver then uses the signature to verify the DNS response, ensuring that the record has not been tampered with. In addition, it also provides a chain of trust from the TLD to the domain authority zone to ensure that the entire DNS resolution process is secure.

Despite these obvious benefits, the adoption of DNSSEC has been slow. The main problem is that DNSSEC settings are very complicated, and it is necessary to upgrade equipment and systems, as well as corresponding services, to process the new protocol. In addition, due to the DNS cache poisoning and other forms of attacks have not received corresponding attention. DNSSEC has not been raised to a higher priority for implementation.

4.3. DDoS Attack

One type of attack directed at the DNS server software itself is a DDoS attack, which typically exploits bugs in the BIND software application to crash or refuse service to the DNS server. The DNS server is not another target of the attack, rather it uses the DNS server as an intermediary ”attack amplifier” that targets other Internet hosts to allow the host to reject the service.

The DDoS attack is shown in Figure 8.

There are two main types of DDoS attacks: bandwidth exhaustion attacks and resource exhaustion attacks. Several methods can be used to contain these two types of attacks effectively:

1) Monitor incoming network traffic. In this way, users can know who is accessing the network, monitor abnormal visitors, and analyze logs and source IP after the fact. The intruder will use a limited number of attacks prior to a large-scale attack to assess the robustness of the network of victims.

2) Use high-performance load balancing software, multiple servers, and deploy them in different data centers.

3) Optimize resource usage to boost the webserver's load power.

4) Use a highly scalable DNS device to protect against DDOS attacks against DNS. For example, some commercial solutions can provide DDOS attack protection against DNS or TCP/IP from 3 to 7 layers.

5) Disable ICMP on the router and only open ICMP when testing is needed.

6) To stop comprehensive DDOS attacks, pay attention to the security setup of the server. For bandwidth-consuming attacks, one of the most effective solutions is to buy more bandwidth.

Knowing the common defense methods of DDoS, let's discuss how to protect traditional authoritative DNS servers from DDoS attacks. The default requests are based on UDP for authoritative DNS, and the DNS protocol explicitly specifies that the services offered by TCP can be limited by the DNS server. Therefore, the most important thing in the defense of DDoS attacks of authoritative DNS is how to prevent UDP attacks. But the biggest problem with UDP DDoS defense is that UDP has no session. If a request is an assault by packet interaction is difficult to decide. It is

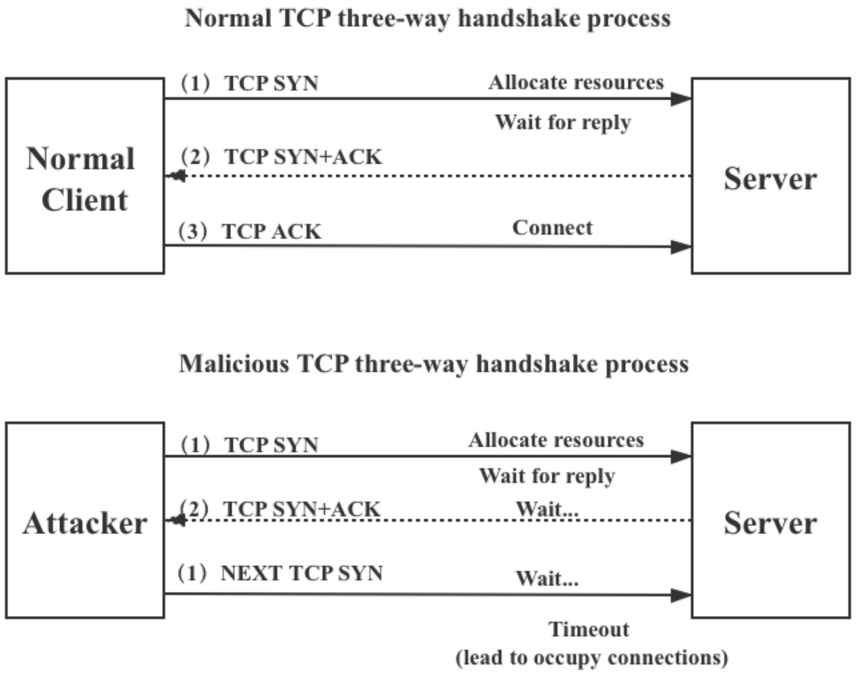


Figure 8. DDoS attack

impossible to distinguish whether a request is an attack or a real user request by just viewing a certain DNS data packet. . Therefore, the primary work of traditional security technologies is to convert UDP back-and-forth requests that lack session interactions into UDP multiple-back and multiple requests with session records. They will use the functionality of the DNS protocol to protect the following technologies:

1) CNAME retransmission: Using DNS features, the recursive request has iterative query attributes before the final response is received. It directly replaces the DNSserver and returns a forged unique random string cname domain name to the client, and determines whether the source IP continues to initiate the request for the cname domain name. If a standard request is an IP. Obviously, the IP will be trusted if an IP automatically initiates a cname domain name request; relatively, if an IP does not initiate a cname domain name request within the specified timeout duration, the IP can then be judged as an intruder.

2) TC retransmission: Using the characteristics of DNS, when the DNS request client encounters the TC flag of the DNS response flag field, it will inevitably initiate a TCP DNS request. It directly replaces the DNS server and returns a forged empty response to the client, but the response flag field is marked with TC It is 1, and it is determined whether the IP is a normal request according to whether the source IP continues to initiate the DNS request for the domain name. Obviously, if an IP immediately initiates a TCP DNS request, then the IP can be trusted; relatively, if an IP does not initiate a targeted TCP request within the specified timeout period, the IP Will be judged as an attacker.

3) First packet discarded: Using the characteristics of DNS, when the DNS request client does not receive a DNS response within the timeout period, it will resend the request. Traditional security directly discards the first packet request, and continues to initiate a second request for this domain name according to the source IP. To decide whether a standard request is an IP. Clearly, the IP will be trusted if an IP initiates a second request in a targeted way; relatively, if an IP does not initiate a second request during the defined time-out span, the IP can be judged as an intruder.

From the above information, we can know that the principles of these three methods are by converting the original DNS UDP one-and-one request into a UDP multiple-request with session records, and by judging the characteristics of the second request. Determine if the source IP is the permission action or attack behavior of a real user, and then execute the corresponding procedure of whitelist/blacklist.

Can the above traditional solutions completely protect authoritative DNS? There are still some security concerns, in truth. Below are summary of the possible problems encountered by authoritative DNS protection:

1) First of all, in terms of first packet discarding, this is a technology that has not been adopted in authoritative DNS defense. The main reason is that recursive DNS will select other authoritative servers based on the SRTT algorithm when an authoritative query request is discarded, resulting in traditional security Basically, the so-called "second request" cannot be received, so the probability of manslaughter is extremely high. At the same time, authoritatively discarding recursively sent queries will have a serious impact on the resource occupation of the recursive server. In this case, the recursive server may directly discard normal requests for the domain name according to its own protection strategy, which may cause more serious failures.

2) The second is TC retransmission. Compared with the CNAME retransmission strategy, the main advantage of TC retransmission is that there is no tampering with the data content information, and there is no "forgery" corresponding response; and the major defect is that it requires a secure service on the DNS server side. Supporting TCP requests is a very big test in performance, and the risk of being paralyzed will increase. In addition, the LocalDNS of some ISPs does not support TCP at all is also an important issue.

3) Let's talk about CNAME retransmission. The biggest problem with CNAME retransmission is that it "forged" a fictitious response. In the normal process, this "forged" response only serves as the result of the intermediate transmission and will not have any other impact. In reality, the various "cache recursive separation" and "cache acceleration response" technologies on the ISP side will tamper with the normal process, causing the aforementioned "forged" result to be treated as the correct result and directly returned to the end user; Worse still, various DNS "optimized TTL" technologies on the ISP side will seriously amplify this problem and eventually lead to serious failures.

How to prevent authoritative DNS? The excellent performance of the DNS system itself is very important. In principle, the traditional security strategy of converting the UDP back and forth request without session interaction to the UDP multiple back and forth policy with session records consumes more computing resources than simply replying to a DNS response. For example, under the same performance conditions and resources, replying to a so-called "cname reply" or "tc reply" is better than replying directly to the native DNS reply. A rough comparison shows that there is no difference in the CPU instruction set between the two. The most critical assumption, of course, is that the DNS system must have outstanding performance, high capacity and the potential to be equal to or even better than a stable server.

4.4. DNS Spoofing

DNS spoofing is a type of fraud that pretends to be a domain name server by an attacker that sometimes occurs before the DNS cache expires. If a record already resides in the DNS cache, the DNS server will return the record directly to the cache as soon as a client query happens.

Users can prevent DNS spoofing in five aspects.

1) Bind IP address and MAC address. DNS spoofing attacks are to change or pretend to be the IP Address of the DNS Server. Therefore, static binding of MAC Address and IP Address can also be used to prevent DNS spoofing. Moreover, the deception of DNS attacks must start with ARP spoofing. If ARP spoofing can be effectively prevented or avoided, it will make DNS ID spoofing attacks impossible.

2) Use Digital Password for identification. The Work Digital Signature (TSIG) technologies may be used to define and validate the data exchange mechanism in the file data transfer of multiple subnets to avoid the outbreak of information leakage or tampering. Also it is extremely difficult to mask the identity of the master-slave server due to the password authentication process, which enhances the reliability of the transmission of domain name information.

3) Optimize the related project settings of DNS SERVER.

4) Use the IP address directly.

5) Monitor DNS data packets. In a DNS spoofing attack, the Client will receive at least two DNS data response packets, one is a real data packet, and the other is an attack data packet. To reply to the client before the real response packet, the spoofing attack data packet has a very simple information data structure compared with the real data packet, with only the response domain. Therefore, the DNS response packets can be monitored, and the corresponding principles and model algorithms can be followed to distinguish these two response packets which help users to avoid the attack of false data packets.

4.5. DNS Amplification Attack

DNS amplification attack takes advantage of the difference in bandwidth consumption between the attacker and the target web resource. Since each robot requires a spoofed IP address to open the DNS resolver, the IP address has been changed to the true source IP address of the target victim. The target will then receive a response from the DNS resolver. Therefore the attacker builds the request in a manner that produces a response from the DNS resolver as much as possible to produce a significant volume of traffic. As a consequence, the target experiences an amplification of the original traffic of the attacker, and bogus traffic blocks their network, resulting in a denial of service.

The precaution of DNS amplification attack can be concluded into four parts.

1) Configure the firewall and network capacity correctly.
2) Increase the link bandwidth.
3) Limit the DNS resolver to only respond to queries from trusted sources or close the recursive queries of the DNS server.
4) Use DDoS defense products to filter, clean abnormal access requests at the entrance and distribute normal access requests to the server for business processing.

Also, It is important to check the configuration of each protocol of the server and disable services that may be used by attack tools.

As for the defence of DNS amplification attack, one of a useful method is to reduce the total number of open DNS resolvers.

An important part of DNS amplification attacks is access to open DNS resolvers. If there is an improperly configured DNS resolver on the Internet, the attackers will easily use these vulnerabilities to implement attacks – they only needs to find this DNS resolver to use it. Ideally, DNS resolver services should be provided only to the device from the trusted domain. For the reflection-based attacks, open DNS resolver will respond to queries anywhere on the Internet, so it could be exploited. Limit the DNS resolver to make it only responds to a query from a trusted source, so that the server can not be used for any type of amplification attacks.

Another method is to verify the IP source to avoid spoofed packets leaving the networks. Because attackers must put the victims'IP address as the destination address to sent the UDP request.Therefore, to reduce the effectiveness of UDP-based amplification attacks, the key is that the Internet service provider (ISP) rejects any internal traffic with deceptive IP addresses. If a packet is transmitted from the internal network, but the source address from the network appears to the outside, it may be spoofed packets, which may be discarded.

5. Future Research Directions

For various security issues of DNS, although a large number of solutions have emerged, various attacks in recent years have shown that DNS security issues are still very serious. Through analysis, it is found that the existing research results still have shortcomings, and future research work can pay more attention to the following aspects.

5.1. Research on Decentralized DNS System

The reason why the DNS system is subject to various attacks is related to the DNS tree structure and root server management of the entire system. This architecture has a single point of failure, and there have been multiple attacks on the root server in history, causing the entire DNS service to be paralyzed. Therefore, designing a decentralized DNS system is an important research topic . Currently, the design of decentralized DNS mainly includes the following two directions.

1) Based on blockchain technology. The emergence of blockchain technology provides technology and framework for decentralized design, and the use of blockchain technology to design a decentralized DNS system is also a new research idea. Currently, the main challenges facing this approach include:
    a) Efficient P2P network design. The problems faced by P2P networks include being vulnerable to network fluctuations. In an environment with severe network fluctuations, query efficiency will be greatly reduced; data forgery and lack of access control make some false information spread to P2P networks . Designing an efficient and secure P2P network is an important subject of decentralized DNS design based on blockchain.
    b) Efficient consensus algorithm design. At present, the main consensus algorithms include methods based on proof of work and proof of rights. However, the consensus efficiency of these consensus algorithms is low, and there will be forks in the consensus process, and they are not suitable for real-time update and maintenance of DNS data. The design of the consensus algorithm needs to be combined with the design of

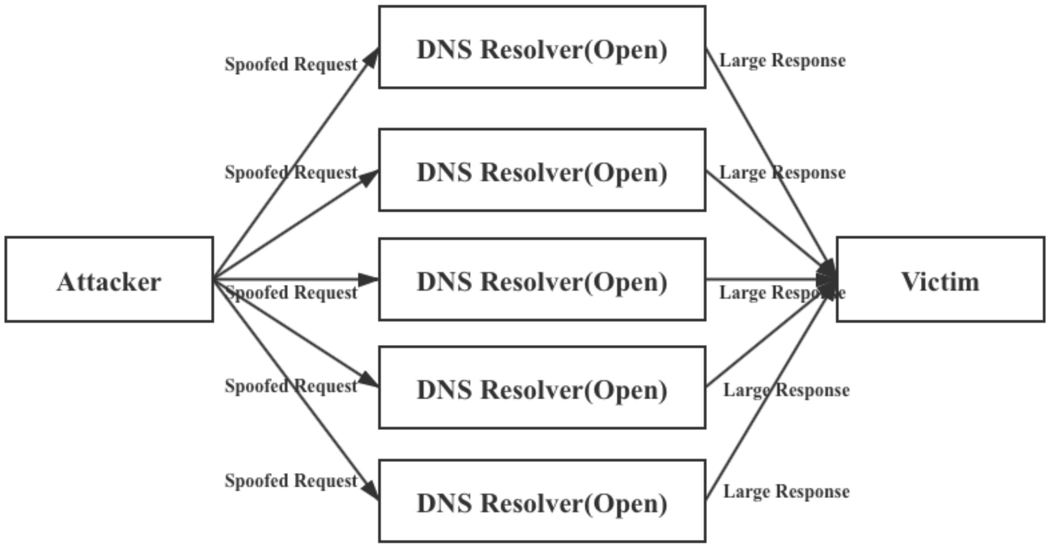


Figure 9. DNS amplification attack

DNS application scenarios. Consensus efficiency should be high and strong consistency should be ensured in the consensus process, and the data stored in each node is consistent.

2) Based on the root server alliance. The main research idea of this scheme is to reduce the centralized control of root servers through alliances of different countries or different top servers. The challenges faced by this approach include the design of the national root alliance architecture, the control of the root alliance system, and the synchronization of the root alliance and the main root server.

5.2. Open DNS Server Security Detection

Although open servers provide various conveniences, such as being able to respond to DNS requests from external resources, these open systems bring great hidden dangers to the security and stability of the network. Some open servers are easily controlled by attackers and carry out malicious actions such as amplification attacks and poisoning attacks. According to the survey, 28 million of the 32 million open parsers have serious security risks. Openness will bring convenience for attackers to carry out DoS/DDoS, buffer poisoning, DNS ID hijacking and other attacks. Few existing researches have standardized and studied these open systems. How to identify and monitor these malicious open servers is also an important research topic. The main challenges faced before include:

1) Most of the existing detection systems have a single detection object. In actual applications, a detection system cannot be configured to detect a certain malicious behavior. Therefore, it is necessary to build a comprehensive detection system that can effectively monitor various security threats.

2) At present, whether based on machine learning, information entropy, or statistical analysis and other theoretical methods, it is difficult to resist special types of attacks, such as attacks initiated by software zero-day vulnerabilities. The detection efficiency of existing detection systems needs to be improved. For detections such as DNS tunnels and privacy leaks, they cannot be discovered in time when information is leaked. Therefore, new detection systems must also meet the timeliness requirements.

5.3. Incremental Deployment of Protection Schemes

Due to the widespread application of the DNS system, although some scholars have proposed improvements, it is still not compatible with the existing DNS system, and it is difficult to be deployed on a large scale. Although DNSSEC was proposed in 1997, it has not yet been widely deployed. As of December 2016, although the deployment rate of DNSSEC in the top-level domain has reached 89%, the deployment rate in the second-level domain is only 3% . Many new name service systems and architectures have been proposed, but they are still not compatible with the current DNS system. Therefore, these research results are difficult to be adopted by network operators and large companies. Therefore, when designing the deployment method of the protection scheme, it should be considered that the protection scheme should avoid modifying the existing DNS protocol.

5.4. DNS Security Detection and Privacy Protection in the Cloud Environment

Internet infrastructure is rapidly shifting to a hybrid private/public cloud model. Cloud services are being widely used, and 93% of organizations use software, infrastructure or platforms as their service objects. Although cloud services have brought great convenience, 42% of organizations are still attacked by cloud application downtime attacks directly from DNS. Such attacks include public clouds and private clouds. In the cloud environment, DNS security faces the following problems.

1) DNS security detection in cloud environment. At present, DNS detection is mainly for the traffic between DNS servers under ISP, and there are few research results on detection of DNS servers in the cloud environment. But in recent years, DNS security under the cloud environment is very severe. Therefore, how to design and detect DNS security issues under cloud services is an explorable direction.

2) DNS-based privacy leak detection in the cloud environment. The leakage of user privacy data in cloud environment is very serious [86], such as the transmission and theft of private data by means of DNS tunnel. How to detect DNS data privacy leakage in the cloud environment is also worthy of attention.

6. Conclusion

DNS is an important infrastructure of the Internet. From the early enhancement of DNS protocol security to the improvement of the current system structure, its security protection has always been a concern of academic and industrial circles. This article provides a comprehensive analysis and summary of DNS security protection technology, and at the same time introduces possible future research hotspots to provide references for further research.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] Hansen T, Hallambaker P. DomainKeys identified mail (DKIM) service overview. Baker, 2009.

[3] Tom L. Improving performance on the Internet. Communications of the ACM, 2009,52(2):44–51.

[4] Levine J. DNS blacklists and whitelists. IETF RFC 5782, 2010.

[5] Almeida VAF, Doneda D, Abreu JDS. Cyberwarfare and digital governance. IEEE Internet Computing, 2017,21(2):68–71.

[6] Chinese CN. https://www.computerworld.com/article/2484097/internet/major-ddos-attacks-cn-domain

[7] Turkey DNS. https://blog.radware.com/security/2015/12/turkey-dns-servers-under-attack/

[8] Bortzmeyer S. DNS privacy considerations. RFC 7626, 2015.

[9] Framework POS. https://www.anomali.com/blog/three-month-frameworkpos-malware-campaign-nabs-43000-credits-cards-from-poi

[10] Research RFC. https://www.rfc-editor.org/search/rfc-search-detail.

[11] UDNS threat survey 2017. 2017.http://www.efficientip.com/resources/white-paper-dns-security-survey-2017/

[12] Anstee D, Bowen P, Chui CF, Sockrider G. In: Proc. of the 12th Worldwide Infrastructure Security Report. Arbor Network, 2017.

[13] Marc K, Hupperich T, Rossow C, et al. Exit from Hell? Reducing the impact of amplification DDoS attacks. In: Proc. of the USENIX. USENIX Association, 2014. 111–125.

[14] Cisco 2017 Midyear Cybersecurity Report. https://www.cisco.com /c/m/en-au/products/security/offers/cybersecurity-reports

[15] Wang Y, Hu M, Li B. Survey on domain name system security. Journal on Communications, 2007,28(9):91–103.

[16] Hu N, Deng P, Yao S, et al. Issues and challenges of Internet DNS security. Chinese Journal of Network and Information Security, 2017,3(3):13–21 (in Chinese with English abstract).

[17] Jiang J. Research on inconsistent and multiple dependence in the authorization mechanism of Internet domain name system [Ph.D. Thesis]. Tsinghua University, 2013 (in Chinese with English abstract).

[18] Liu Q. The security of Internet domain name system in China. Modern Telecommunications Technology, 2010,2010(4):9–11 (in Chinese with English abstract).

[19] Li J. Detection of DNS spoofing and cache poisoning attacks [MS. Thesis]. Chengdu: University of Electronic Science and Technology of China, 2015 (in Chinese with English abstract).

[20] Schomp K, Callahan T, Rabinovich M, et al. Assessing DNS vulnerability to record injection. In: Proc. of the Int'l Conf. on Passive and Active Measurement. 2014. 214–223.

[21] Mohaisen A. Evaluation of privacy for DNS private exchange. IETF Internet Draft, 2015-05.

[22] Bortzmeyer S. DNS privacy considerations. IETF RFC 7626, 2015.

[23] Rossebo J, Cadzow S, Sijben P, et al. A threat, vulnerability and risk assessment method and tool for Europe. In: Proc. of the Int'l Conf. on Availability, Reliability and Security. 2007. 925–933.

[24] Banse C, Herrmann D, Federrath H. Tracking users on the Internet with behavioral patterns: Evaluation of its practical feasibility. In: Information Security and Privacy Research. Berlin, Heidelberg: Springer-Verlag, 2012. 235–248.

[25] Ariyapperuma S, Mitchell CJ. Security vulnerabilities in DNS and DNSSEC. In: Proc. of the 2nd Int'l Conf. on Availability, Reliability and Security. 2007. 335–342.

[26] Schomp K, Callahan T, Rabinovich M, et al. On measuring the client-side DNS infrastructure. In: Proc. of the Conf. on Internet Measurement. 2013. 77–90.

[27] Callahan T, Allman M, Rabinovich M. On modern DNS behavior and properties. ACM SIGCOMM Computer Communication Review, 2013,43(3):7–15.

[28] Shulman H, Waidner M. Towards security of Internet naming infrastructure. In: Proc. of the Computer Security-ESORICS. 2015.

[29] DNS Server Software Distribution. https://ftp.isc.org/www/survey/reports/2017/07/fpdns.txt

[30] Bind security vulnerabilities. CVE-2019-6465, 2019.

[31] Microsoft DNS server vulnerability. https://support.microsoft.com/en-us/help/2678371/microsoft-dns-server-vulnerability-to-dnsserver-cache-snooping-attack

[32] Learn more at National vulnerability database (NVD). CVE-2017-11779, 2017.

[33] Microsoft Windows DNS server denial of service vulnerability. https://tools.cisco.com/security/center/viewAlert.x?alertId=53604

[34] Microsoft Windows DNS server cache poisoning vulnerability. https://www.securityfocus.com/bid/30132/

[35] Yeti DNS project. https://yeti-dns.org/

[36] Ateniese G, Mangard S. A new approach to DNS security (DNSSEC). In: Proc. of the 8th ACM Conf. on Computer and Communications Security. 2001. 86–95.

[37] Yang H, Osterweil E, Massey D, Lu SW, Zhang LX. Deploying cryptography in Internet-scale systems: A case study on DNSSEC. IEEE Trans. on Dependable and Secure Computing, 2011,8:656–669.

[38] Herzberg A, Shulman H. DNSSEC: Interoperability challenges and transition mechanisms. In: Proc. of the 7th Int'l Conf. on Availability, Reliability and Security. 2013. 398–405.

[39] Herzberg A, Shulman H. DNSSEC: Security and availability challenges. In: Proc. of the Communications and Network Security. 2013. 365–366.

[40] Lian W, Rescorla E, Shacham H, et al. Measuring the practical impact of DNSSEC deployment. In: Proc. of the USENIX Security. 2013. 573–588.

[41] Dempsky M. DNSCurve: Link-level security for the domain name system. Internet Draft draft-dempsky-dnscurve-01, RFC, 2010.

[42] Anagnostopoulos M, Kambourakis G, Konstantinou E, Gritzalis S. DNSSEC vs. DNSCurve: A side-by-side comparison. In: Proc. of the IGI Global. 2012. 201–220.

[43] Zhu L, Hu Z, Heidemann J, et al. Connection-oriented DNS to improve privacy and security. In: Proc. of the ACM Conf. on SIGCOMM. 2015. 379–380.

[44] Shulman H. Pretty bad privacy: Pitfalls of DNS encryption. In: Proc. of the Workshop on Privacy in the Electronic Society. 2014. 191–200.

[45] Park K, Pai VS, Peterson L, et al. CoDNS: Improving DNS performance and reliability via cooperative lookups. In: Proc. of the 6th Symp. on Operating Systems Design and Implementation. San Francisco, 2004. 14.

[46] Poole L, Pai VS. ConfiDNS: Leveraging scale and history to improve DNS security. In: Proc. of the 3rd Workshop on Real, Large Distributed Systems (WORLDS). 2006.

[47] Khurshid A, Kiyak F, Caesar M. Improving robustness of DNS to software vulnerabilities. In: Proc. of the 27th Annual Computer Security Applications Conf. Orlando, 2011. 177–186.

[48] Huang K, Kong N. Research on status of DNS privacy. Computer Engineering and Applications, 2018,54(9):28–36 (in Chinese with English abstract).

[49] Scaife N, Carter H, Traynor P. OnionDNS: A seizure-resistant top-level domain. In: Proc. of the Communications and Network Security. 2015. 379–387.

[50] Herrmann D, Fuchs K, Lindemann J, et al. EncDNS: Alightweight privacy-preserving name resolution service. In: Proc. of the Computer Security-ESORICS 2014. 2014. 37–55.

[51] Choi H, Lee H, Kim H. BotGAD: Detecting botnets by capturing group activities in network traffic. In: Proc. of the 4th Int'l ICST Conf. on Communication System Software and Middleware. ACM, 2009. 2.

[52] Choi H, Lee H. Identifying botnets by capturing group activities in DNS traffic. Computer Networks, 2012,56(1):20–33.

[53] Babak R, Perdisci R, Antonakakis M. Segugio: Efficient behavior-based tracking of malware-control domains in large ISP networks. In: Proc. of the Int'l Conf. on Dependable Systems and Networks. IEEE, 2015. 403–414.

[54] Perdisci R, Corona I, Dagon D, Lee W. Detecting malicious flux service networks through passive analysis of recursive DNS traces. In: Proc. of the Annual Computer Security Applications Conf. (ACSAC 2009). IEEE, 2009. 311–320.

[55] Huang SY, Mao CH, Lee H M. Fast-flux service network detection based on spatial snapshot mechanism for delay-free detection. In: Proc. of the 5th ACM Symp. on Information, Computer and Communications Security. 2010. 101–111.

[56] Yadav S, Reddy AN. Winning with DNS failures: Strategies for faster botnet detection. In: Rajarajan M, Piper F, Wang H, Kesidis G, eds. Security and Privacy in Communication Networks. Berlin, Heidelberg: Springer-Verlag, 2012. 446–459.

[57] Dong LP, Chen XY, Yang YJ, et al. Implementation and detection of network covert channel. Computer Science, 2015,42(7): 216–244 (in Chinese with English abstract).

[58] Antonakakis M, Perdisci R, Dagon D, et al. Building a dynamic reputation system for DNS. In: Proc. of the USENIX Security. 2010. 18–36.

[59] Bilge L, Kirda E, Kruegel C, et al. EXPOSURE: Finding malicious domains using passive DNS analysis. In: Proc. of the Network and Distributed System Security Symp. (NDSS 2011). 2011.

[60] Antonakakis M, Perdisci R, Lee W, et al. Detecting malware domains at the upper DNS hierarchy. In: Proc. of the 20th USENIX Conf. on Security. 2011. 21–27.

[61] Bishop CM. Pattern Recognition and Machine Learning (Information Science and Statistics). New York: Springer-Verlag, 2006.

[62] Zhang WW, Gong J, Liu SD, Hu XY. DNS surveillance on backbone. Ruan Jian Xue Bao/Journal of Software,2017,28(9): 23702387 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5186.htm [doi: 10.13328/j.cnki.jos.005186]

[63] Dabek F, Kaashoek MF, Karger D, Morris R, Stoica I. Wide-area cooperative storage with CFS. In: Proc. of the ACM Symp. on Operating Systems Principles (SOSP 2001). Chateau Lake Louise, 2001.

[64] Stoica I, Morris R, Karger D, et al. Chord: A scalable peer-to-peer lookup service for Internet applications. IEEE/ACM Trans. on Networking, 2003,11(1):17–32.

[65] Cox R, Muthitacharoen A, Morris R. Serving DNS using a peer-to-peer lookup service. In: Proc. of the 1st Int'l Workshop on Peer-to-Peer Systems. Cambridge, 2002. 155–165.

[66] Maymounkov P, Mazières D. Kademlia: A peer-to-peer information system based on the XOR metric. In: Proc. of the 1st Int'l Workshop on Peer-to-Peer Systems (IPTPS 2001). London: Springer-Verlag, 2002. 53–65.

[67] Danielis P, Altmann V, Skodzik J, Wegner T, Koerner A, Timmermann D. P-DONAS: A P2P-based domain name system in access networks. ACM Trans. on Internet Technology, 2015,15(3):11.

[68] Ramasubramanian V, Sirer EGU. Beehive: O(1) lookup performance for power-law query distributions in peer-to-peer overlays. In: Proc. of the 1st Conf. on Networked Systems Design and Implementation. San Francisco, 2004.

[69] Ramasubramanian V, Sirer EGU. The design and implementation of a next generation name service for the Internet. In: Proc. of the 2004 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications. 2004. 331–342.

[70] Song Y, Koyanagi K. Study on a hybrid P2P based DNS. In: Proc. of the IEEE Int'l Conf. on Computer Science and Automation Engineering. Shanghai, 2011. 152–155.

[71] Tsai WT, Yu L, Wang R, Liu N, Deng EY. Blockchain application development techniques. Ruan Jian Xue Bao/Journal of Software, 2017,28(6):14741487 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5232.htm [doi: 10.13328/j.cnki.jos.005232]

[72] Namecoin. https://Namecoin.info

[73] Satoshi N. Bitcoin: A peer-to-peer electronic cash system. 2009.

[74] Ali M, Nelson J, Shea R, et al. Blockstack: A global naming and storage system secured by block chains. In: Proc. of the 2016 USENIX Annual Technical Conf. (USENIX ATC 16). 2016. 181–194.

[75] ENS. https://ens.domains/

[76] PeerName. https://peername.com/

[77] EMCDNS. https://emercoin.com/

[78] Eyal I, Sirer EG. Majority is not enough: Bitcoin mining is vulnerable. In: Proc. of the Financial Cryptography. 2014.

[79] Simplified name verification protocol. https://blockstack.org/

[80] Zhang Y, Xia CD, Fang BX, et al. An autonomous open root resolution architecture for domain name system in the Internet. Journal of Cyber Security, 2017,2(4) (in Chinese with English abstract).

[81] Fang BX. Discussion on autonomous root domain name system based on national union from "Network Sovereignty". Information Security and Communications Privacy, 2014(12):35–38 (in Chinese with English abstract).

[82] Zhu GK, Jiang WB. A decentralized domain name system for the network. Cyberspace Security, 2017,8(1):14–18 (in Chinese with English abstract).

[83] Lamport L. The part-time parliament. ACM Trans. on Computer Systems, 1998,16(2):133–169.

[84] Open resolver project. 2016. http://openresolverproject.org/

[85] Lu ZH, Gao XH, Huang SJ, et al. Scalable and reliable live streaming service through coordinating CDN and P2P. In: Proc. of the Int'l Conf. on Parallel and Distributed Systems. IEEE, 2012. 581–588.

[86] DNSSEC deployment report. http://rick.eng.br/dnssecstat/

[87] Bhadauria R, Sanyal S. Survey on security issues in cloud computing and associated mitigation techniques. arXiv Preprint arXiv:1204.0764, 2012.

[88] Botnet in ddos attacks:trends and challenges. Hoque N,Bhattacharyya D K,Kalita J K. IEEE Communications Surveys and Tutorials . 2015

[89] Diameter of the world wide web. R Albert, H Jeong, A-L Barabasi. Nature . 1999

[90] DDoS With The Slow HTTP POST Attack. Kelly Jackson Higgins. Dark Reading . 2010

[91] Botz-4-sale: surviving organized DDoS attacks that mimic flash crowds. S. Kandula,D. Katabi,M. Jacob, et al. Proceedings of the 2nd Symposium on Networked Systems Design Implementation (NSDI"05) . 2005

[92] Understanding the Mirai Botnet. Antonakakis M,April T,Bailey M,et al. 26th USENIX Security Symposium . 2017

[93] Low-Rate and High-Rate Distributed Do SAttack Detection Using Partial Rank Correlation. Bhuyan M H,Kalwar A,Goswami A,et al. Fifth International Conference on Communication Systems and Network Technologies . 2015

[94] Detecting TCP-based DDoS attacks in Baidu Cloud Computing Data Centers. Jiao J H,Ye B J,Zhao Yet al. 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS) . 2017

[95] Learning to control a structured-prediction decoder for detection of HTTP-layer DDo S attackers. Dick U,Scheffer T. Machine Learning . 2016

[96] A study on Web security incidents in China by analyzing vulnerability disclosure platforms[J] . Cheng Huang,JiaYong Liu,Yong Fang,Zheng Zuo. Computers Security . 2016

[97] A Novel Method to Defense Against Web DDoS[J] . Yan Haitao,Wang Fengyu,Cao ZhenZhong,Lin Fengbo,Chen Chuantong. International Journal of Digital Content Technolo . 2012 (19)

[98] A taxonomy of DDoS attack and DDoS defense mechanisms[J] . Jelena Mirkovic,Peter Reiher. ACM SIGCOMM Computer Communication Review . 2004 (2)

[99] DNS Zone Transfer Protocol (AXFR). HOENES A,LEWIS E. IETF RFC5936 . 2010

[100] Mitigating Client Subnet Leakage in DNS Queries. PAN L,ZHANG X,HU A,et al. 2018 16th Annual Conference on Privacy, Security and Trust (PST). IEEE Computer Society, 16th Annual Conference on Privacy, Security and Trust . 2018

[101] IoT SP"18-Web-based Attacks to Discover and Co ntrol Local Io T Devices. Acar G,Huang D Y,Li F,et al. Proceedings of the 2018 Workshop on IoT Security and Priv acy . 2018

[102] Cross-Origin Resource Sharing COR S. Mozilla. https://developer.mozilla.org/en-US/docs/Web/HTT P/CORS . 2019

[103] Dynamic pharming attacks and the locked same-origin policies for web browsers. C.Karlof,U.Shankar,J.Tygar,D.Wagner. The 14th ACM Conference on Computer and Communication Security (CCS"07) . 2007

[104] Attacking Private Networks from th e Internet with DNS Rebinding. Brannon Dorsey. DEFCON . 2018

[105] Eradica ting DNS rebinding with the extended same-origin policy. Martin Johns,Sebastian Lekies,Ben Stock. Proceedings of the 22nd USENIX conference .

[106] FireDrill:interactive DNS rebinding. Yunxing Dai,Ryan Resig. Proceedings of the 7th USENIX conference on Offensive Technologies . 2013

[107] The Web Origin Concept. Lt A B,Gt I. . 2011

[108] Protecting browsers from DNS rebinding attacks. Jackson, Collin,Barth, Adam,Bortz, Andrew,Shao, Weidong,Boneh, Dan. ACM Transactions on the Web . 2009

[109] Born Kenton and Gustafson David 2010 Detecting dns tunnels using character frequency analysis arXiv preprint arXiv: 1004.4358

[110] Lin Huaqing, Liu Gao and Yan Zheng 2019 International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage (Cham: Springer) Detection of application-layer tunnels with rules and machine learning

[111] Buczak Anna L. 2016 Detection of tunnels in PCAP data by random forests Proceedings of the 11th Annual Cyber and Information Security Research Conference

[112] Kara A. Mert 2014 Detection of malicious payload distribution channels in DNS 2014 IEEE International Conference on Communications (ICC). IEEE

[113] Qi Cheng 2013 A bigram based real time DNS tunnel detection approach Procedia Computer Science 17 852-860

[114] Yu Bin 2016 Behavior Analysis based DNS Tunneling Detection and Classification with Big Data Technologies IoTBD

[115] Liu Jingkun 2017 Detecting DNS tunnel through binary-classification based on behavior features 2017 IEEE Trustcom/BigDataSE/ICESS. IEEE

[116] Mugali Aditya Anand, Simpson Andrew W. and Walker Scott King 2015 System and method for detecting DNS traffic anomalies U.S. Patent No. 9, 172, 716. 27 Oct.

[117] Davis Jonathan J. and Foo Ernest 2016 Automated feature engineering for HTTP tunnel detection Computers Security 59 166-185

[118] You-Qiang L. U. O. 2017 DNS tunnel Trojan detection method based on communication behavior analysis Journal of ZheJiang University (Engineering Science) 51 1780-1787

[119] Nadler Asaf, Aminov Avi and Shabtai Asaf 2019 Detection of malicious and low throughput data exfiltration over the DNS protocol Computers Security 80 36-53

[120] Jianqiang Yang and Hongxi J. 2016 Using FQDN number of the second-level domain name to detect DNS-based covert channels Computer Era

[121] Herrmann Dominik, Banse Christian and Federrath Hannes 2013 Behavior-based tracking: Exploiting characteristic patterns in DNS traffic Computers Security 39 17-33

[122] Kirchler Matthias 2016 Tracked without a trace: linking sessions of users by unsupervised learning of patterns in their DNS traffic Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security

[123] Hao Shuai and Wang Haining 2017 Exploring domain name based features on the effectiveness of dns caching ACM SIGCOMM Computer Communication Review 47 36-42

[124] Wang Yun-Yun and Chen Song-Can 2011 A survey of evaluation and design for AUC based classifier Pattern Recognition and Artificial Intelligence 1

[125] Research on DDoS Defense Based on DNS Response Value Evaluation[J]. Yue Qiaoli, Lu Wanbo, Hu Weihong, Zhang Haikuo. Information Network Security. 2019(09)

[126] The Security Challenge of DDoS to DNS[J]. He Kun. Network Security Technology and Application. 2010(01)

[127] Arends, Roy, et al. DNS security introduction and requirements. RFC 4033 (Proposed Standard), 2005.

[128] Ariyapperuma, Suranjith, and Chris J. Mitchell. ”Security vulnerabilities in DNS and DNSSEC.” The Second International Conference on Availability, Reliability and Security (ARES’07). IEEE, 2007.

[129] Herzberg, Amir, and Haya Shulman. ”Security of patched DNS.” European Symposium on Research in Computer Security. Springer, Berlin, Heidelberg, 2012.

[130] Osterweil, Eric, Dan Massey, and Lixia Zhang. ”Deploying and monitoring dns security (dnssec).” 2009 Annual Computer Security Applications Conference. IEEE, 2009.