Journal of Multidisciplinary Engineering Science and Technology (JMEST)
ISSN: 2458-9403
Vol. 6 Issue 6, June - 2019

Design And Implementation Of Security System For A Domain Name Server (DNS) Using Cryptography

Dr. Udeh Chukwuma (Ph.D)
Director Information & Communication Technology (ICT)
Enugu State University of Technology Teaching Hospital (ESUTTH), Enugu
uchukwuma14@yahoo.com; +2348033653812.

Abstract: Design and implementation of Domain Name Sever (DNS) using Cryptography has been investigated. The DNS commonly means to verify whether the data comes from the authorized system. Weakness in security through DNS cache, DNS spoofing or due to weak authentication during the server exchanging updates became common that the need for DNNSEC was considered imperative. Due to this problem, the study aimed at designing DNS Security by combining the concept of both the Digital Signature and Asymmetric key (Public key) Cryptography to provide security for the domain name. Here the Public key is send instead of Private key. The DNS security uses Message Digest Algorithm to compress the Message (text file) and PRNG (Pseudo Random Number Generator) Algorithm for generating Public and Private key. The message combines with the Private keys to form a Signature using DSA Algorithm, which is sent along with the Public key. The receiver uses the Public key and DSA Algorithm to form a Signature. If this Signature matches with the Signature of the message received, the message is Decrypted and read else discarded.

Keywords- design, implementation, cryptography, DNS (Domain name server), public key, private key.

1. Introduction

DNS is the standard mechanism for name to IP address resolution [1]. The Domain Name System is a protocol for locating domain names and mapping them to IP addresses [2]. DNS is a hierarchical, distributed database, which provides mapping between easy to remember hostnames, such as www.mdurohtak.ac.in, and IPv4 or IPv6 network addresses, for example, 117.211.115.134. For practical security and availability reasons it is important that DNS is able to tolerate failures and attacks. This is evident from recent phishing attacks [3] that have used DNS cache poisoning to steal sensitive financial data [4].

The Domain Name System (DNS) translates the Internet domain and host names to IP addresses and vice versa. DNS converts the names typed in our Web browser address bar to the IP addresses of Web servers of sites. Many companies use DNS to manage their personal network. Networks at homes use DNS when accessing the Internet. DNS clients send requests and receive responses from DNS servers respectively. Requests that contain a name, that result in an IP address being returned from the server, are called forward DNS lookups while requests that contain an IP address resulting in a name are called reverse DNS lookups.

The Domain Name System (DNS) has become a critical operational part of the Internet Infrastructure, yet it has no strong security mechanisms to assure Data Integrity or Authentication. To solve the known security problems with DNS, a set of security extensions (DNSSEC) have been proposed [5]. DNSSEC provides data integrity and origin authentication using pre-generated digital signatures for each data item stored in the DNS database. These Digital Signatures are included in the zones as resource records. The extensions also provide for the storage of Authenticated Public keys in the DNS. This storage of keys can support general Public key distribution services as well as DNS security. These stored keys enables security aware resolvers to learn the authenticating key of zones, in addition to those for which they are initially configured.

However, Cryptography is the science of "Secret Writing". The process of changing the plain information called the plaintext into some sort of code called the cipher text is Encryption. The reverse of Encryption is Decryption. Cryptography includes various methods for providing security like Symmetric Key Cryptography, Public Key Cryptography and the Elliptic Curve Cryptography. Cryptography has also been expanded to provide several information security requirements which include Non-repudiation (Preventing an entity from denying previous commitments or actions), Integrity (Ensuring no unauthorized alteration of data), Authentication (Verifying an entity's identity) and Confidentiality (Protecting the data from all but the intended receiver).

The Keys associated with DNS names can be retrieved to support other protocols. In addition, the security extensions provide for the Authentication of DNS protocol transactions. The DNS Security is designed to provide security by combining the concept of both the Digital Signature and Asymmetric key (Public key) Cryptography. Here the Public key is send instead of Private key. The DNS security uses Message Digest Algorithm to compress the Message (text file) and PRNG(Pseudo Random Number Generator) Algorithm for generating Public and Private key. The message combines with the Private key to form a Signature using DSA Algorithm, which is send along with the Public key. The receiver uses the Public key and DSA Algorithm to form a Signature. If this Signature matches with the Signature of the message received, the message is Decrypted and read else discarded.

At most times, DNS undergoes distributed denial of service (DDoS) attacks, and also reflection and amplification attacks. Because many companies/organisations use only a couple of DNS servers, DNS security can be easily bypassed by a volumetric attack, causing DNS servers to go offline and preventing users from accessing the website. So, DNS Security is very essential in our everyday lives as we must protect the data from attacks and eliminate all vulnerabilities. In this study we considered it obvious to use cryptography (DES encryption) to implement the proposed model. In DES, the same key is used to encrypt and decrypt a message, so both the sender and the receiver should know and use the same private key. The DES is a block cipher, which means that a cryptographic key and algorithm are applied to a block of data one bit at a time rather than simultaneously. For a plaintext message to be encrypted, DES groups it into 64-bit blocks. Every block is en-ciphered using the secret key into a 64-bit cipher text using permutation and substitution. This process involves 16 rounds and can run in four various modes, by encrypting blocks individually or making every cipher block dependent on all their previous blocks. Decryption is simply the reverse of encryption, where the same steps are followed but reversing the order in which the keys are applied. The most basic method of attack for any cipher is brute force, which involves trying each key until you find the right one.

However, as indicated in this study, DNSSEC introduces new security issues such as chain of trust problems, timing and synchronization attacks, Denial of Service amplification, increased computational load, and a range of key management issues.

2. Review of Related Literature

The DNS security is designed to provide security through the concept of both the digital signature and the asymmetric (public) key cryptography. The public and the private key are both used by the receiver and the sender respectively. To intensify the security to the DNS to face these security problems, the IETF included some security extensions to the DNS, generally referred to as DNSSEC [7]. The Domain Name System is a protocol that is used for mapping the domain names to protocols. Elliptic Curve Cryptography (ECC) is a public key. Elliptic Curve Digital Signature Algorithm (ECDSA); a cryptographic algorithm used by financial institutions to ensure that funds can only be spent by their rightful owners. It is fast at verifying the signatures and uses small key size when compared to RSA and provides same level of security as given by RSA [8]. The analysis of network traffic data is considered for identifying largely supple patterns in DNS. If any change is observed, it is mapped onto the bipartite graph which is then checked for cybercrime [9]. The quantum key distribution (QKD) security relies on the no-cloning

theorem, which allows not copying a quantum system properly. An efficient tool is suggested to study the quantum channel activity and arrange the properties of a quantum cloning-based attack for DV and CVQKD protocols [10]. Cryptography-based, prefix-preserving anonymous technique is developed that is provably as secure as the well-known TCP dpriv scheme and unlike TCPdriv, provides a consistent prefix-preservation in large scale distributed setting and evaluation of the security properties inherent in all prefix-preserving IP address anonymization schemes (including TCPdpriv) [11]

Caching is done in networks for data access in the internet to be made easier. It is highly required for DNS. DNS is usually built with resource records which contains mapping of all names and addresses. When the reference cannot be done locally (client), the request is put to the server which then replies back to the client. All these caches have a time limit (TTL) [12]. However, when this is done, the basic design of e-mail is vulnerable and can easily be attacked. The major threats are spam, phishing and denial of service (DoS) while the prevention methods are the Sender Policy Framework (SPF) [13], The Sender-ID Framework (SIDF) [14] and The Domain Keys Identified Mail (DKIM) design [15].

Nowadays vulnerability is found in almost all systems and hence system administrators must prioritize them in accordance to the vulnerability. So, to prevent exploitation all stakeholders must be ready to set ACL scripts to deny access to intruders [17].Two key protocols are analysed state-aware and state-less protocols and also analyze security-goals and threats respectively [18].

These procedures analyses the threats that hamper cloud computing implementation on a large scale and also about the services offered by the current vendors. Both system administrators and client users while reading this work would also get to know about the future of cloud security research and equally know about the security risks associated with it [19]. The major reason for the business world's indisposition to use the internet as a viable source of communication is the fear of security gaps. So they prefer cryptographic measures for developing internet's infrastructure over physical segregation. However, the need for the introduction of IPSec architecture with security protocols in the network layer along with the transport layer security protocols [20]. Also to keep the vulnerabilities from exploitation of the hosts server and the network an improved method for network security which consists of the network management, the vulnerability scan, the risk assessment, the access control and the incident notification has been introduced [21, 22]. For the scope of this research, the researcher considered three email sender authentication mechanisms (i.e.) DNS : SPF, DKIM and Sender - ID Framework which are designed to assist in filtering of all undesirable mails in particular spam and phishing mails [23]. And to focus on the anonymizing the IP addresses in the trace, a cryptographic based prefix preserving anonymization technique is used while the security properties of all prefix preserving anonymization schemes were also evaluated [24]. Primarily taking into consideration the importance of the consumers in the online business world, a secured flexible and a cost effective cloud computing services are provided [25]. Optical cryptography based on computational ghost imaging (CGI) which is the major procedure that converts plaintext into a random intensity vector is a simplification of transformation and it is vulnerable too [26].

2.1 Conceptual clarifications

2.1.1 Information Leakage

A successful zone transfer by an attacker may serve as an investigation attack, potentially revealing sensitive information about internal network configuration, e.g. the IP addresses of internal firewall interfaces. DNS names could, for instance, represent project names that may be of interest to an attacker, or however could reveal the identity of the operating system running on the machine.

2.1.2 Dynamic Update Vulnerabilities

Protocols like Dynamic Host Configuration Protocol (DHCP) make use of the DNS Dynamic Update protocol to add and delete RRs on demand. These updates take place on the primary server of the zone [12]. The authentication for such updates is based solely on source IP address, which is vulnerable to threats such as IP spoofing. These attacks range from denial of service, including deletion of records, to redirection [21].

2.1.3 BIND Security Considerations

DNS servers across the Internet using the BIND implementation of DNS software are not constantly up to date with security patches and software updates. As a result, some time a significant fraction of the Internet's DNS servers are vulnerable to compromise [27]. The majority of these vulnerabilities are a result of poor exception handling and boundary checking. Exploitation will potentially allow attackers to execute arbitrary code or write data to arbitrary locations in memory [11].

2.1.4 Usage vulnerabilities

The use of DNS triggers conditions similar to DDoS attacks. The DNS query rate at the root servers has risen from 1 query per second to roughly 100000 queries per second in 2004 [17, 25]. Measurements at root servers show an amazing number of bogus queries: from 60-85 % of observed queries were repeated from the same host within the measurement interval. Over 14 % of a root server's query load is due to queries that violate the DNS specification [8].

Root servers receive a large number of queries be-cause the resolvers never receive the replies, because of either packet filters or routing issues [32]. Improper us-age or improper coding of the client resolvers can cause a DDoS effect on DNS root servers.

2.1.5 Domain Name System Security Extensions (DNSSEC)

DNSSEC adds security to the DNS protocol by providing authentication, data integrity and authenticated denial of existence to DNS data provided by a name server. All answers from DNSSEC servers are digitally signed. By checking the signature, a DNSSEC resolver is able to check if the information originated from a legitimate server and that data is identical to the data on the authoritative DNS server. If the data is not present on the server an authenticated denial is produced.

To maintain backward compatibility with DNS, DNSSEC requires only minor changes to the DNS protocol. DNSSEC adds four record types to DNS, namely Resource Record Signature (RRSIG), DNS Public Key (DNSKEY), Delegation Signer (DS), and Next Secure (NSEC). DNSSEC uses two of the previously unused flag bits in the DNS query and answer message header (AD and CD). The AD (Authentic Data) bit in a response indicates that all the data included in the answer and authority portion of the response has been authenticated by the server. The CD (checking disabled) bit indicates that unauthenticated data is acceptable to the resolver sending the query [5]. Because the UDP protocol has a packet size limit of 512 bits, DNSSEC requires the use of EDNS0 [29] extensions that override this limitation, so that larger key sizes can be accommodated [10].

DNSSEC adds the ability to detect MITM attacks on DNS through the addition of data origin authentication, transaction and request authentications, but DNSSEC does not prevent such attacks. To maintain data origin authenticity and integrity, both servers and resolvers must use the DNSSEC protocol.

2.1.6 Keys in DNSSEC

Each secured zone has a key pair, made up of a zone private key and the corresponding public key. The zone public key is stored as a resource record (type KEY) in the secured zone. The public key is used by DNS servers and Resolvers to verify the zone's digital signature.

All resource records in a secured zone are signed by the zone's private key. To make zone resigning and key roll overs easier to implement, it is possible to use one or more keys as Key Signing Keys (KSKs). A KSK will only be used to sign the top level KEY RRs in a zone. Zone Signing Keys (ZSKs) are used to sign all the RRsets in a zone.

2.1.7 Signatures in DNSSEC

DNSSEC provides an unforgeable authentication of an RRset by associating it with a signature resource record that binds DNS data to a time interval and the signer's domain name. A private key is used to sign an RRset. For increased speed a hash of the RRset is signed. This provides authenticated data origin. If data is modified during transport the signature is no longer valid (authenticated data integrity). In DNSSEC, only signatures are used, and nothing is encrypted.

Hashes are generated using MD5 or SHA-1. Signatures are created using MD5/RSA [4], DSA [3] or elliptic curve cryptographic algorithms. Signatures are

stored as resource records (type RRSIG) and are used with the zone's public key to authenticate resource records.

2.1.8 Time in DNSSEC

All times in DNS are relative. The Start of Authority (SOA) resource record's refresh, retry and expiration timers are counters that are used to determine the time elapsed since a slave server synchronised with a master server. The Time to Live (TTL) value is used to determine how long a forwarder should cache data after it has been fetched from an authoritative server. DNSSEC introduces absolute time into DNS.

The signature validity period is the period that a sig-nature is valid. It starts at the time specified in the sig-nature inception field of the RRSIG and ends at the time specified in the expiration field. The signature publication period is the time after which a signature is replaced with a new signature by publishing the relevant RRSIG in the master zone file.

2.1.9 DNSSEC Vulnerabilities

DNSSEC does not guard against poor configuration or bad information in the authoritative name server, and does not protect against buffer overruns or DDoS attacks. A large increase in the computational load on the servers and resolvers, a hierarchical model of trust, the lack of management tools, and the need for a higher level of time synchronisation between the servers, re-main some of the most significant obstacles to its deployment.

Cryptographic key management issues, such as initial key configuration and key rollover, key authentication and verification have yet to be resolved at an operational level in order to enable DNSSEC to be deployed on a global scale. Storage of the zone private key is also an issue, and DNSSEC cannot tolerate malicious server failures. Finally as SECREG [12] test bed experience suggests, DNSSEC is complex to implement.

2.2 Statement of the Problem

Original DNS specifications did not include security based on the fact that the information that it contains, namely host names and IP addresses, is used as a means of communicating data [SPAF]. As more and more IP based applications developed, the trend for using IP addresses and host names as a basis for allowing or disallowing access (i.e., system based authentication) grew. Unix saw the advent of Berkeley "r" commands (e.g., rlogin, rsh, etc.) and their dependencies on host names for authentication. Then many other protocols evolved with similar dependencies, such as Network File System (NFS), X windows, Hypertext Transfer Protocol (HTTP), et al.

Another contributing factor to the vulnerabilities in the DNS is that the DNS is designed to be a public database in which the concept of restricting access to information within the DNS name space is purposely not part of the protocol. Later versions of the BIND implementation allow access controls for such things as zone transfers, but all in all, the concept of restricting who can query the DNS for RRs is considered outside the scope of the protocol.

The existence and widespread use of such protocols as the r-commands put demands on the accuracy of information contained in the DNS. False information within the DNS can lead to unexpected and potentially dangerous exposures. The majority of the weaknesses within the DNS fall into one of the following categories: Cache poisoning, client flooding, dynamic update vulnerability, information leakage, and compromise of the DNS server's authoritative database.

3. Methodology

The DES (Data Encryption Standard) algorithm was adopted and used in this study. For many years, and among many people, "secret code is making" and DES have been synonymous. DES works on bits, or binary numbers--the 0s and 1s common to digital computers. Each group of four bits makes up a hexadecimal, or base 16, number. Binary "0001" is equal to the hexadecimal number "1", binary "1000" is equal to the hexadecimal number "8", "1001" is equal to the hexadecimal number "9", "1010" is equal to the hexadecimal number "A", and "1111" is equal to the hexadecimal number "F".

DES works by encrypting groups of 64 message bits, which is the same as 16 hexadecimal numbers. To do the encryption, DES uses "keys" where are also *apparently* 16 hexadecimal numbers long or *apparently* 64 bits long. However, every 8th key bit is ignored in the DES algorithm, so that the effective key size is 56 bits. But, in

any case, 64 bits (16 hexadecimal digits) is the round number upon which DES is organized.

3.1 Existing System

Authenticity is based on the identity of some entity. This entity has to prove that it is genuine. In many Network applications the identity of participating entities is simply determined by their names or addresses. High level applications use mainly names for authentication purposes, because address lists are much harder to create, understand, and maintain than name lists.

Assuming an entity wants to spoof the identity of some other entity, it is enough to change the mapping between its low level address and its high level name. It means that an attacker can fake the name of someone by modifying the association of his address from his own name to the name he wants to impersonate. Once an attacker has done that, an authenticator can no longer distinguish between the true and fake entity.

3.2 Proposed System

Taking the above prevailing system into consideration the best solution is using Pseudo Random Number Generator for generating Key Pair in a quick and more secured manner. We use MD5 (or) SHA-1 for producing Message Digest and Compressing the message. Signature is created using Private Key and Message Digest which is transmitted along with the Public Key. The transfer of the packets from each System to System is shown using Graphical User Interface (GUI). Each time the System get the message, it verifies the IP Address of the sender and if no match is found it discards it. For verification, the Destination System generates Signature using Public Key and DSA Algorithm and verifies it with received one. If it matches it Decrypts otherwise it discards. The new system would be fast and efficient to work, ease of access to system and less manual effort would be required.

For example, if we take the plaintext message "7878787878787878", and encrypt it with the DES key "0E329232EA6D0D73", we end up with the cipher text "0000000000000000". If the cipher text is decrypted with the same secret DES key "0E329232EA6D0D73", the result is the original plaintext "7878787878787878".

This example is neat and orderly because our plaintext was exactly 64 bits long. The same would be true if the plaintext happened to be a multiple of 64 bits. But most messages will not fall into this category. They will not be an exact multiple of 64 bits (that is, an exact multiple of 16 hexadecimal numbers).

3.3 High level model of the new system

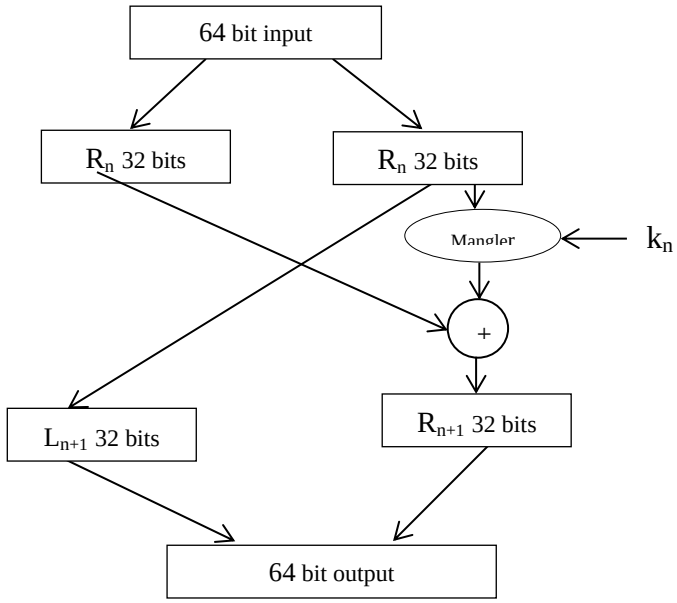

Fig.1: The above image depicts the working of DES Encryption.

The programming languages used were java, C and C++ while the Remote Method Invocation (RMI) was deployed as an Application Program Interface (API) that helps in the study to create a distributed application in Java by providing some mechanism/process. The RMI accepts the request of objects to appeal methods on another object which is currently working/running in a different Java Virtual Machine (JVM).

Procedure: If the study adopts the plaintext message

"Your lips are smoother than Vaseline" is, in hexadecimal, "596F7572206C6970 732061726520736D 6F6F746865722074 68616E2076617365 6C696E650D0A". (Note here that the first 72 hexadecimal digits represent the English message, while "0D" is hexadecimal for Carriage Return, and "0A" is hexadecimal for Line Feed, showing that the message file has terminated.)

Then padding this message with some 0s on the end, to get a total of 80 hexadecimal digits: "596F7572206C6970 732061726520736D 6F6F746865722074 68616E2076617365 6C696E650D0A0000". If we then encrypt this plaintext message 64 bits (16 hexadecimal digits) at a time, using the same DES key "0E329232EA6D0D73" as before, we get the cipher text: "C0999FDDE378D7ED 727DA00BCA5A84EE 47F269A4D6438190 9DD52F78F5358499 828AC9B453E0E653". This is the secret code that can be transmitted or stored. Decrypting the cipher text restores the original message "Your lips are smoother than Vaseline".

3.4 How the New System Works

Data Encryption Standard (DES) is a block cipher which operates on plaintext blocks of a given size (64-bits) and returns cipher text blocks of the same size. Thus DES results in a permutation among the 2exp64 (read as: "2 to the power of 64") possible arrangements of 64 bits, each of which may be either 0 or 1. Each block of 64 bits is sub-divided into two blocks of 32 bits each, a left half block P and a right half Q. (This division is only used in certain operations.)

Example: Let P be the plain text message, P = 0123456789ABCDEF, where P is in hexadecimal (base 16) format. Rewriting P in binary format, we get the 64-bit block of text:

P = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

Q = 0000 0001 0010 0011 0100 0101 0110 0111

Then;

Q = 1000 1001 1010 1011 1100 1101 1110 1111

The first bit of P is "0". The last bit is "1". However, reading is done from left to right.

DES operates on the 64-bit blocks using key sizes of 56- bits. The keys are actually stored as being 64 bits long, but every 8th bit in the key is not used (i.e. bits numbered 8, 16, 24, 32, 40, 48, 56, and 64). However, the number nevertheless are in bits and must be numbered from 1 to 64, going left to right, in the following calculations. But, in doing so the eight bits just mentioned get eliminated when we create sub keys.

Example: Let n be the hexadecimal key n = 133457799BBCDFF1. This gives us as the binary key (setting 1 = 0001, 3 = 0011, etc., and grouping together every eight bits, of which the last one in each group will be unused): n = 00010011 00110100 01010111 01111001 10011011 10111100 11011111 11110001 The DES algorithm uses the following steps:

3.4.1 Algorithm:

Get the key, usually a 64-bit key as input from user in order to have a correct parity, each byte should contain "1" bits in odd numbers. Every 8th bit is a parity bit). Calculate the key schedule by performing a permutation on the 64-bit key. (The parity bits are removed, making the key to 56 bits. 1st bit of the permuted block is bit 57 of the original key, 2nd bit is bit 49, and 56th bit is bit 4 of the original key.)

Split the key after permutation into two halves. The first 28 bits can be called E[0] and the 28 bits from last are called F[0]. Then, Calculate 16 sub-keys starting with i = 1. Find either one or two circular left shifts on both E[i-1] and F[i-1] to get E[i] and F[i], respectively. Find the permutation for the concatenation E[i]F[i]. This will give n[i], which is 48 bits long.

However, Loop back to E[i-1] and F[i-1] until n[16] has been calculated. Processing a 64-bit data block as input from the user. If the block is shorter than 64 bits, it should be considered as appropriate. Permutation on the data blocks and split after permutations into two halves to produce a two equal 32 bits ciphers. The first 32 bits in the block are called Q[0], and the last 32 bits are called R[0].

Apply the 16 sub-keys that are found on the data block. Starting with i = 1. Make the 32-bit R[i-1] into 48 bits by expanding. Perform XOR for E(R[i-1]) with K[i].the, split E(R[i-1]) n[i] into eight 6-bit blocks. Bits 1-6 are B[1], bits 7-12 are B[2], and so on with bits 43-48 as B[8].

Again, substitute those values in the S-boxes for all B[j]. Starting with j = 1. All the values in the S-boxes should be considered 4 bits wide. Consider the 1st and 6th bits of B[j] together as a 2-bit value (call it m) indicating the row in S[j] to look for substitution. Take the 2nd-5th bits of B[j] together as a 4-bit value (call it n) indicating the column in S[j] to find substitution. Replace B[j] with value

of S[j][m][n] and loop back from B[j] until all 8 blocks have been replaced.

Finally, carry out permutation for the concatenation of B[1] through B[8] and XOR the resulting value with Q[i-1]. Thus, your R[i] = Q[i-1] P(S[1](B[1])...S[8](B[8])), where B[j] is a 6-bit block of E(R[i-1]) K[i]. (The function for R[i] is written as, R[i] = Q[i-1] f(R[i-1], K[i]).)

When:

L[i] = R[i-1]; then loop back from *R[i-1]* until K[16] has been applied. Carry out permutation on the block R[16]L[16] to get:

Qn+1 = Rn, and

Rn+1 = Qn ^{P}Nk (Rn)

In DNS, this encryption is applied to secure the outgoing/incoming traffic from a client device to the DNS server, while the reverse of this algorithm performs decryption. Vulnerabilities in the DNS have frequently been exploited for attacks on the Internet. One of the most common ways of "defacing" a web server is to redirect its domain name to the address of a host controlled by the attacker through manipulation of the DNS. DNSSEC [9] eliminates some of these problems by providing end-to-end authenticity and data integrity through transaction signatures and zone signing.

Transaction signatures are computed by clients and servers over requests and responses. DNSSEC allows the two parties either to use a message authentication code (MAC) with a shared secret key or public-key signatures for authenticating and authorizing DNS messages between them. The usefulness of transaction signatures is limited since they guarantee integrity only if a client engages in a transaction with the server being authoritative for the returned data, but do not protect against a corrupted server acting as a resolver. For zone signing, a public-key for a digital signature scheme, called a zone key, is associated with every zone. Every resource record (it is the basic data unit in the DNS database) is complemented with an additional SIG resource record containing a digital signature, computed over the resource record.

Careful generation of all keys is a sometimes overlooked but absolutely essential element in any cryptographically secure system. The strongest algorithms used with the longest keys are still of no use if an adversary can guess enough to lower the size of the likely key space so that it can be exhaustively searched. Technical suggestions for the generation of random keys will be found in RFC 4086 [14]. One should carefully assess if the random number generator used during key generation adheres to these suggestions.

Keys with a long effectively period are particularly sensitive as they will represent a more valuable target and be subject to attack for a longer time than short period keys. It is strongly recommended that long-term key generation occur off-line in a manner isolated from the network via an air gap or, at a minimum, high-level secure hardware.

4. Summary and Conclusions

DNS has major security issues that need to be addressed urgently. Threats such as man in the middle attacks and cache poisoning arise because of the lack of authentication and integrity in the DNS transaction process. Usage threats are caused by a range of entities from misconfigured client resolvers to packet filters causing conditions similar to DDoS. The Internet Engineering Task Force (IETF) are really responding to the threats by developing DNSSEC, a secure DNS protocol, to address the data integrity and source spoofing issues. DNSSEC allows transaction level authentication and secure zone transfers protecting all data in the zone during the transfer. In DNSSEC, Name-based authentication attacks can be detected [7].

DNSSEC does not protect against buffer overruns or DDoS attacks, nor does it provide confidentiality. Secure delegation is complex to implement, and DNSSEC's error reporting capabilities are minimal. DNSSEC zone files are significantly larger than their DNS counterparts, including the load on servers, net-work and resolver.

The public/private keys used with DNSSEC can be compromised over time. Keys should be changed at intervals to reduce the risk of compromise. This can be implemented relatively easily at the lower levels of the DNSSEC hierarchy, as the public keys are not cached for very long. Root key rollover however is a problem, as authentication (chain of trust) is based on known root keys.

Regular key rollover has a significant impact on the entire DNSSEC structure. Selection of timing parameters is critical in DNSSEC, involving TTL, signature inception time and signature expiration time.

Despite the vulnerabilities in DNSSEC, it provides integrity and authentication to DNS data. DNSSEC is backward compatible with the existing DNS infrastructure. The study however, suggest further research in areas such as advanced dynamic zone transfer protocols using link state type algorithms, unification of alternate routes to form a meta root using the ICANN root, resiliency of DNSSEC, active attacks against distributed directory services such as X.500, amongst others to improve the core functionality of DNSSEC.

5. References

[1]. Partridge, C. and Trewitt, G. (2010). HEMS variable definitions. RFC 1024, Internet Engineering Task Force.
[2]. Postel, J. B. (2015). TCP and IP bake off. RFC 1025, Internet Engineering Task Force.
[3]. Leyden, J. (2006). Phishing morphs into pharming. Technical report, www.theregister.co.uk
[4]. Ollmann, G. (2005). The phishing guide. Technical report, Next Generation Security Software (NGS).
[5]. Arends, R., Austein, R., Larson, M. Massey, D. and Rose. S., (2005). DNS security introduction and requirements. RFC 4033, Internet Engineering Task Force.
[6]. Suranjith, A. and Chris J. M., (2015). Security vulnerabilities in DNS and DNSSEC. Information Security Group Royal Holloway, University of London. Egham, Surrey TW20 0EX, UK.
[7]. Lalith, A., Sai Gopal P., Ashwath, A. L A. , Vignesh, M. and Lavanya K., (2017). Security system for DNS using cryptography. International Journal of Software and Web Sciences (IJSWS); 1(20), 12-18.
[8]. Thajoddin, S. and Khan, A., (2017). Security System for DNS Using Cryptography. International Journal and Magazine of Engineering, Technology, Management and Research; 4(5), 191-194.
[9]. D. E. 3rd. Secure domain name system dynamic update. RFC 2137, Internet Engineering Task Force, Apr. 1997.
[10]. D. E. 3rd. DNS security operational considerations. RFC 2541, Internet Engineering Task Force, Mar. 1999.
[11]. D. E. 3rd. DSA KEYs and SIGs in the domain name system (DNS). RFC 2536, Internet Engineering Task Force, Mar. 1999.
[12]. D. E. 3rd. Secure domain name system dynamic update. RFC 2137, Internet Engineering Task Force, Apr. 1997.
[13]. D. E. 3rd. DNS security operational considerations. RFC 2541, Internet Engineering Task Force, Mar. 1999.
[14]. D. E. 3rd. DSA KEYs and SIGs in the domain name system (DNS). RFC 2536, Internet Engineering Task Force, Mar. 1999.
[15]. D. E. 3rd. RSA/MD5 KEYs and SIGs in the domain name system (DNS). RFC 2537, Internet Engineering Task Force, Mar. 1999.
[16]. R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS security introduction and requirements. RFC 4033, Internet Engineering Task Force, Mar. 2005.
[17]. D. Atkins and R. Austein. Threat analysis of the domain name system (DNS). RFC 3833, Internet Engineering Task Force, Aug. 2004.
[18]. S. M. Bellovin, J. Schiller, and C. Kaufman. Security mechanisms for the Internet. RFC 3631, Internet Engineering Task Force, Dec. 2003.
[19]. N. Brownlee, K. Claffy, and E. Nemeth. DNS measurements at a root server. Technical report, Cooperative Association for Internet Data Analysis — CAIDA, San Diego Supercomputer Center, University of California, San Diego, 2001.
[20]. D. W. Chadwick and G. Zhao, editors. Public Key Infrastructure, Second European PKI Workshop: Research and Applications, EuroPKI 2005, Canterbury, UK, June 30 - July 1, 2005, Revised Selected Papers, volume 3545 of Lecture Notes in Computer Science. Springer, 2005.
[21]. B. Cohen. DNSSEC: Security for Essential Network Services. Enterprise Networking Planet, 2003.
[22]. I. A. Finlay. CERT advisory CA-2002-15 denial-of-service vulnerability in ISC BIND 9. Internet, 2002.
[23]. R. Gieben. DNSSEC in NL Final Report. Technical report, NLnet Labs, 2004.
[24]. L. Grangeia. Dns cache snooping. Technical report, Securi Team — Beyond Security, February 2004.
[25]. D. E. 3rd. Secure domain name system dynamic update. RFC 2137, Internet Engineering Task Force, Apr. 1997
[26] D. E. 3rd. DSA KEYs and SIGs in the domain name system (DNS). RFC 2536, Internet Engineering Task Force, Mar. 1999.
[27]. D. E. 3rd. RSA/MD5 KEYs and SIGs in the domain name system (DNS). RFC 2537, Internet Engineering Task Force, Mar. 1999.
[28]. R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS security introduction and requirements. RFC 4033, Internet Engineering Task Force, Mar. 2005.
[29]. N. Brownlee, K. Claffy, and E. Nemeth. DNS measurements at a root server. Technical report, Cooperative Association for Internet Data Analysis — CAIDA, San Diego Supercomputer Center, University of California, San Diego, 2001.
[30]. D. W. Chadwick and G. Zhao, editors. Public Key Infrastructure, Second European PKI Workshop: Research and Applications, EuroPKI 2005, Canterbury, UK, June 30 - July 1, 2005, Revised Selected Papers, volume 3545 of Lecture Notes in Computer Science. Springer, 2005.
[31]. B. Cohen. DNSSEC: Security for Essential Network Services. Enterprise Networking Planet, 2003.
[32]. Reseaux IP Europeens (RIPE) NCC. K-root statistics, k.root-servers.net. Technical report, Reseaux IP Europeens, 2004.