GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Security System For DNS Using Cryptography

A report for the evaluation 3 of project 2

Submitted by

MANISHA SINGH

(1613101375/16SCSE101130)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Dr. J.N. Singh

Professor.

APRIL/MAY-2020

TABLE OF CONTENTS

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"SECURITY SYSTEM FOR DNS USING CRYPTOGRAPHY"** is the bonafide work of "**MANISHA SINGH(1613101375)"** who carried out the project work under my supervision.

SIGNATURE OF HEAD

DR. MUNISH SHABARWAL,
PhD(Management),PhD(CS)
Professor & Dean,
School of computing Science & Engineering

SIGNATURE OF SUPERVISIOR

Dr.J.N Singh,
Professor
School of computing Science & Engineering

Abstract:

To reach another person on the Internet we have to type an address into our computer - a name or a number. That address has to be unique so computers know where to find each other. ICANN coordinates these unique identifiers across the world. Without that coordination we wouldn't have one global Internet. When typing a name, that name must be first translated into a number by a system before the connection can be established. That system is called the Domain Name System (DNS) and it translates names like www.icann.org into the numbers – called Internet Protocol (IP) addresses. ICANN coordinates the addressing system to ensure all the addresses are unique.

Recently vulnerabilities in the DNS were discovered that allow an attacker to hijack this process of looking some one up or looking a site up on the Internet using their name. The purpose of the attack is to take control of the session to, for example, send the user to the hijacker's own deceptive web site for account and password collection.

These vulnerabilities have increased interest in introducing a technology called DNS Security Extensions (DNSSEC) to secure this part of the Internet's infrastructure. DNSSEC will ensure the end user is connecting to the actual web site or other service corresponding to a particular domain name.

INTRODUCTION:

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 aIntegrity or Authentication. Extensions to the DNS are described that provide these services to security aware resolves are applications through the use of Cryptographic Digital Signatures. These Digital Signatures are included 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. 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). 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.

DNS SECURITY PROBLEMS

It is known the fact that DNS is weak in several places. Using the Domain Name System we face the problem of trusting the information that came from a non authenticated authority, the name-based authentication process, and the problem of accepting additional information that was not requested and that may be incorrect. “Many of the classic security breaches in the history of computers and computer networking have had to do not with fundamental algorithm or protocol flaws, but with implementation errors. While we do not intend to demean the efforts of those involved in upgrading the Internet protocols to make security a more realistic goal, we have observed that if BIND would just do what the DNS specifications say it should do, stop crashing, and start checking its inputs, then most of the existing security holes in DNS as practiced would go away.” - Paul Vixie, founder of ISC and main programmer of BIND.

Trusting Supplementary : Non-Authoritative Information

This is another side of the DNS weakness. For the goal of efficiency the DNS was designed to have the additional section in its standard message format. Therefore, in certain cases when a supplementary information is considered necessary to speed up the response for a given query, it is included in the additional section. One example, if we ask our name server, i.e. ns.mydomain.com, to retrieve the mail exchange RR for the domain comp-craiova.ro, the server responds with a mail.gate.comp-craiova.ro RR in the answer section of the DNS message and in the additional section the name and the IP address of the name server authoritative for the comp-craiova.ro domain are included.

Remember that this information was not explicitly asked by us, rather it was cached by our name server, in its pursuit for solving our query, in order to avoid further lookups for the name server authoritative for that domain.

The type of attack possible in this case is called “cache poisoning”. How does this happen? Suppose the following situation described in figure 1. An attacker controlling the name server for his domain evil.com wants to poison the cache of another name server called ns.broker.com used by a broker’s agency in order to impersonate the machine www.bank.com that is often accessed by the users in the domain broker.com. From his machine the attacker asks the name server ns.broker.com for a name under the authority of his own name server, e.g. anyhost.evil.com. The name server ns.broker.com will contact the attacker’s name server - authoritative for that name. This name server will answer the query and will also get the query ID which it stores for later use. This query ID is placed in the header section of any DNS message and is assigned by the pro- gram that generated the query (i.e., the target name server). This identifier is copied in the corresponding reply and can be used by the requester to match up replies to outstanding queries - as mentioned in the RFC 1035 [2]. The at- tack continues with another query from the attacker’s side. He knows that the broker’s agency is frequently contacting a certain bank site whose name he is willing to spoof. Therefore, he will ask the ns.broker.com name server for the address of the www.bank.com. Normally, the name server ns.broker.com will contact the DNS server authoritative for the domain bank.com (e.g., ns.bank.com). At this point, the attacker will start to flood the ns.broker.com server with replies in which the address of the attacker’s machine is mapped to the name www.bank.com before the true response can arrive from the authoritative name server (that is ns.bank.com). He also can predict correctly the query and reply ID, since he already has the last query ID generated by ns.broker.com. In this way, the server ns.broker.com receives an information which is not proper and also caches it after responding to the former attacker’s query for www.bank.com. Now, the trap is set and all the attacker has to do is wait until a connection from broker.com domain is made to www.bank.com. Since the IP address of the attacker’s machine is mapped incorrectly, in the server’s cache (ns.broker.com), to the name www.bank.com all the connections to the bank will be directed to the attacker’s machine. The name server will not try to query again for www.bank.com, it will just use the information it cached during previous DNS lookups. This is another reason why data received by the name servers in the DNS need origin authentication and integrity verifications.

Involving Cryptography

The need for security extensions to DNS was acknowledged and standardized in an organized manner within the DNSSEC IETF working group. The first step is to provide data authentication of the resource records travelling back and forth in the internet. With authentication come also data integrity and data source authentication. The authentication is obtained by means of cryptographic digital signatures. The public key algorithms used for authentication in DNSSEC are MD5/RSA and DSA. The digital signatures generated with public key algorithms have the advantage that anyone having the public key can verify them. Each resource record in the DNS messages exchanged can be digitally signed providing data origin authentication and integrity of the message.

DNSSEC Scope

The scope of the security extensions to the DNS can be summarized into three services: key distribution, data origin authentication, and transaction and request authentication.

Key Distribution

The key distribution service not only allows for the retrieval of the public key of a DNS name to verify the authenticity of the DNS zone data, but it also provides a mechanism through which any key associated with a DNS name can be used for purposes other than DNS. The public key distribution service supports several different types of keys and several different types of key algorithms.

Data Origin Authentication

Data origin authentication is the crux of the design of DNSSEC. It mitigates such threats as cache poisoning and zone data compromise on a DNS server. The RR Sets within a zone are cryptographically signed thereby giving a high level of assuredness to resolvers and servers that the data just received can be trusted.

DNSSEC makes use of digital signature technology to sign DNS RR Set. The digital signature contains the encrypted hash of the RR Set. The hash is a cryptographic checksum of the data contained in the RR Set. The hash is signed (i.e., digitally encrypted) using a private key usually belonging to the originator of the information, known as the signer or the signing authority. The recipient of the RR Set can then check the digital signature against the data in the RR Set just

received. The recipient does this by first decrypting the digital signature using the public key of the signer to obtain the original hash of the data. Then the recipient computes its own hash on the RR Set data using the same cryptographic checksum algorithm, and compares the results of the hash found in the digital signature against the hash just computed. If the two hash values match, the data has integrity and the origin of the data is authentic [CHAR].

Existing System:

The existing system fails when there are rare outcomes or predictors, as the algorithm is based on bootstrap sampling.

The previous results indicate that the stock price is unpredictable when the traditional classifier is used.

The existence system reported highly predictive values, by selecting an appropriate time period for their experiment to obtain highly predictive scores.

The existing system does not perform well when there is a change in the operating environment.

It doesn’t focus on external events in the environment, like news events or social media.

It exploits only one data source, thus highly biased.

Proposed system

The proposed model has following features-

Secure DNS is a big change but inevitable.DNS has been extended to provide security services (DNSSEC) through symmetric-key cryptography. Offers the highest level of security while reducing network traffic. In addition, it reduces storage requirements and enables efficient mutual authentication.

(2)Here proposal for DNSSEC is that, when properly implemented, offers the highest level of security while reducing network traffic. In addition, it reduces storage requirements and enables efficient mutual authentication. DNSSEC can not provide protection against threats from information leakage. This is more of an issue of controlling access, which is beyond the scope of coverage for DNSSEC. Adequate protection against information leakage is already provided through such things as split DNS configuration.

DNSSEC demonstrates some promising capability to protect the Internet infrastructure from DNS based attacks. DNSSEC has some fairly complicated issues surrounding its development, configuration, and management.

PROBLEM STATEMENT

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.

Taking the above prevailing system into consideration the first step is to provide data authentication of the resource records travelling back and forth in the internet. With authentication come also data integrity and data source authentication. The authentication is obtained by means of cryptographic digital signatures. The public key algorithms used for authentication in DNSSEC are MD5/RSA and DSA. The digital signatures generated with public key algorithms have the advantage that anyone having the public key can verify them. Each resource record in the DNS messages exchanged can be digitally signed providing data origin authentication and integrity of the message.

Each time the System get the message, 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 Following functions avoids the pitfalls of the existing system.

- Fast and efficient work
- Ease of access to system
- Manual effort is reduced

Implementation/Architecture diagram

Threats to the Domain Name System

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), etc.

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.

Cache Poisoning

Whenever a DNS server does not have the answer to a query within its cache, the DNS server can pass the query onto another DNS server on behalf of the client. If the server passes the query onto another DNS server that has incorrect information, whether placed there intentionally or unintentionally, then cache poising can occur [CA97]. Malicious cache poisoning is commonly referred to as DNS spoofing [MENM].

- **Cache Poisoning Methods**

Earlier versions of the BIND implementation of the DNS were highly susceptible to cache poisoning. As a means to give a helpful hint, a DNS server responding to a query, but not necessarily with an answer, filled in the additional records section of the DNS response message with information that did not necessarily relate to the answer. A DNS server accepting this response did not perform any necessary checks to assure that the additional information was correct or even related in some way to the answer (i.e., that the responding server had appropriate authority over those records). The naïve DNS server accepts this information and adds to the cache corruption problem. Another problem with earlier versions of BIND is that there wasn‟t a mechanism in place to assure that the answer received was related to the original question. The DNS server receiving the response cache‟s the answer, again contributing to the cache corruption problem. Note that although it is well documented that the BIND implementation has experienced such issues, other implementations may have had, and still may have similar problems.

For example, suppose there is a name server, known as ourdns.example.com, servicing a network of computers (see Figure 5). These computers are in essence DNS clients. An application on a client system, host1, makes a DNS query that is sent to ourdns.example.com. Then ourdns.example.com examines its cache to see if it already has the answer to the query. For purposes of the example, ourdns.example.com is not authoritative for the DNS name in the query nor does it have the answer to the query already in its cache. It must send the query to another server, called brokendns.example.org. The information on brokendns.example.org happens to be incorrect, most commonly due to misconfiguration, and the response sent back to ourdns.example.com contains misleading information. Since ourdns.example.com is caching responses, it caches this misleading information and sends the response back to host1. As long as this information exists in the cache of ourdns.example.com, all clients, not just host1, are now susceptible to receiving this bogus information.

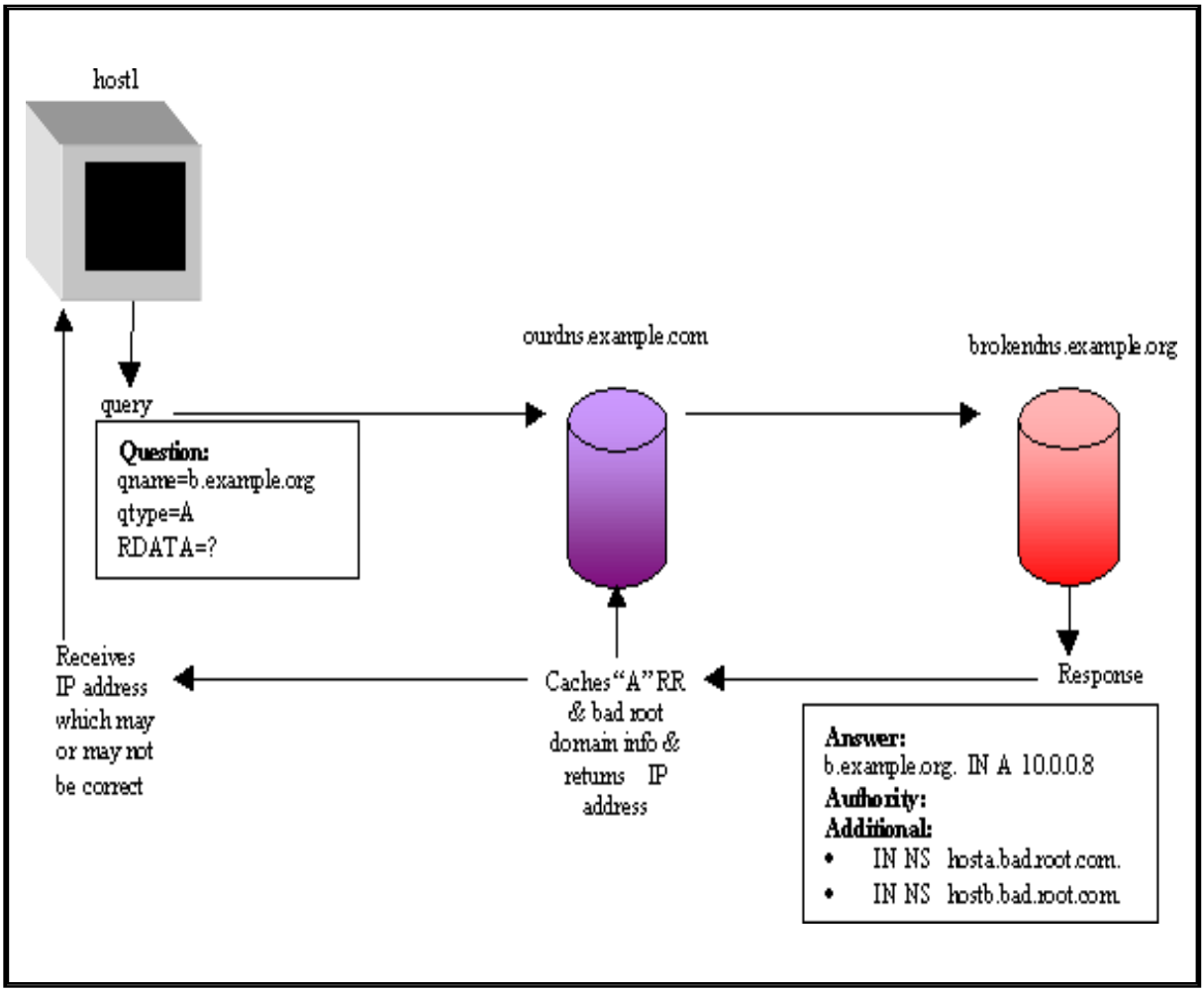

Figure. DNS Cache Poisoning

- **Rogue servers**

Rogue DNS servers pose a threat to the Internet community because the information these servers contain may not be trustworthy [SPAF]. They facilitate attack techniques such as host name spoofing and DNS spoofing. Host name spoofing is a specific technique used with PTR records. It differs slightly from most DNS spoofing techniques in that all the transactions that transpire are legitimate according to the DNS protocol while this is not necessarily the case for other types of DNS spoofing. With host name spoofing, the DNS server legitimately attempts to resolve a PTR query using a legitimate DNS server for the zone belonging to that PTR. It‟s the PTR record in the zone‟s data file on the primary server that is purposely configured to point somewhere else, typically a trusted host for another site [STEV]. Host name spoofing can have a TTL of 0 resulting in no caching of the misleading information, even though the host name is being spoofed. A more detailed example follows later that demonstrates the threats such servers pose to the Internet community.

- **Cache Poisoning Attacks**

An attacker can take advantage of the cache poisoning weakness by using his/her rogue name server and intentionally formulating misleading information. This bogus information is sent as either the answer or as just a helpful hint and gets cached by the unsuspecting DNS server. One way to coerce a susceptible server into obtaining the false information is for the attacker to send a query to a remote DNS server requesting information pertaining to a DNS zone for which the attacker‟s DNS server is authoritative. Having cached this information, the remote DNS server is likely to misdirect legitimate clients it serves [ACME].With earlier versions of the BIND implementation, an attacker can inject bogus information into a DNS cache without the need to worry over whether or not a query was generated to invoke such a response. This willingness to accept and cache any response message allows an attacker to manipulate such things as host name to IP address mappings, NS record mappings, et al. A February 1999 survey revealed that approximately 33% of DNS servers on the Internet are still susceptible to cache poisoning [MENM].This is the methodology used by Eugene Kashpureff. Kashpureff injected bogus information into DNS caches around the world concerning DNS information pertaining to Network Solutions Inc.‟s (NSI) Internet‟s Network Information Center (InterNIC). The information redirected legitimate clients wishing to communicate with the web server at the InterNIC to Kashpureff‟s AlterNIC web server. Kashpureff did this as a political stunt protesting the Internic‟s control over DNS domains. When the attack occurred in July of 1997, many DNS servers were injected with this false information and traffic for the Internic went to AlterNIC where Kashpureff‟s web page was filled with the propaganda surrounding his motives and objections to InterNIC‟s control over the DNS [RAFT].

- **Attack Objectives**

An attacker makes use of cache poisoning for one of two reasons. One is a denial of service (DoS) and the other is masquerading as a trusted entity.

- **Denial of Service**

DoS is accomplished in several ways. One takes advantage of negative responses (i.e., responses that indicate the DNS name in the query cannot be resolved). By sending back the negative response for a DNS name that could otherwise be resolved, results in a DoS for the client wishing to communicate in some manner with the DNS name in the query. The other way DoS is accomplished is for the rogue server to send a response that redirects the client to a different system that does not contain the service the client desires.Another DoS associated with cache poisoning involves inserting a CNAME record into a cache that refers to itself as the canonical name.

- **Masquerading**

The second and potentially more damaging reason to poison DNS caches is to redirect communications to masquerade as a trusted entity. If this is accomplished, an attacker can intercept, analyze, and/or intentionally corrupt the communications [CA97]. The misdirection of traffic between two communicating systems facilitates attacks such as industrial espionage and can be carried out virtually undetected [MENM]. An attacker can give the injected cache a short time to live making it appear and disappear quickly enough to avoid detection.Masquerading attacks are possible simply due to the fact that quite a number of IP based applications use host names and/or IP addresses as a mechanism of providing host-based authentication.

Conclusion/Future work:

WORK DONE

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 who is 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.1 Zone signing also protects relayed data because the signature is created by the entity who owns the zone.

Key Generation

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 ggenerator 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.

Encryption and Decryption Signature Creation Signature Verification.

Conclusion

The DNS as an Internet standard to solve the issues of scalability surrounding the hosts.txt file. Since then, the widespread use of the DNS and its ability to resolve host names into IP addresses for both users and applications alike in a timely and fairly reliable manner, makes it a critical component of the Internet. The distributed management of the DNS and support for redundancy of DNS zones across multiple servers promotes its robust characteristics. However, the original DNS protocol specifications did not include security. Without security, the DNS is vulnerable to attacks stemming from cache poisoning techniques, client flooding, dynamic update vulnerabilities, information leakage, and compromise of a DNS server‟s authoritative files.

- In order to add security to the DNS to address these threats, the IETF added security extensions to the DNS, collectively known as DNSSEC. DNSSEC provides authentication and integrity to the DNS. With the exception of information leakage, these extensions address the majority of problems that make such attacks possible. Cache poisoning and client flooding attacks are mitigated with the addition of data origin authentication for RRSets as signatures are computed on the RRSets to provide proof of authenticity. Dynamic update vulnerabilities are mitigated with the addition of transaction and request authentication, providing the necessary assurance to DNS servers that the update is authentic. Even the threat from compromise of the DNS server‟s authoritative files is almost eliminated as the SIG RR are created using a zone‟s private key that is kept off-line as to assure key‟s integrity which in turn protects the zone file from tampering. Keeping a copy of the zone‟s master file off- line when the SIGs are generated takes that assurance one step further.

- DNSSEC can not provide protection against threats from information leakage. This is more of an issue of controlling access, which is beyond the scope of coverage for DNSSEC. Adequate protection against information leakage is already provided through such things as split DNS configuration.

- DNSSEC demonstrates some promising capability to protect the Internet infrastructure from DNS based attacks. DNSSEC has some fairly complicated issues surrounding its development, configuration, and management. Although the discussion of these issues is beyond the scope of this survey, they are documented in RFC 2535 and RFC 2541 and give some interesting insight into the inner design and functions of DNSSEC. In addition to keep the scope of this paper down, many topics such as secure zone transfer have been omitted but are part of the specifications in RFC 2535.

References

1. Albitz, P. and Liu, C., (1997) „DNS and Bind“, 2nd Ed., Sebastopol, CA, O“Reilly &Associates, pp.1- 9

2. HerbertSchildt, Edition (2003) „The Complete Reference JAVA 2“ Tata McGraw Hill Publications

3. IETF DNSSEC WG, (1994) „DNS Security (dnssec) Charter“, IETF.

4. Michael Foley and Mark McCulley, Edition(2002) ‘JFC Unleashed“

Prentice-Hall India.

5. Mockapetris, P., (1987) „Domain Names – Concepts and Facilities“.