Ministry Higher Education and Scientific Research

Kasdi Merbah University Ouargla

Faculty of New Technologies of Information and Communication

Department of Computer Science and Information Technologies

Professional Master's degree

PRESENT TO GET THE DIPLOMA OF MASTER

Specialty: Security and Network Administration.

Domain: Computer and Information Technology

Spinneret: Computer Science

Specialty: Network Administration and Security

Presented by: BENARABI Manel and DAOUD Rima

Theme

Security DNS request using DNSsec approach

Framed by:

DR.BOUKHAMLA Akram.

College Year: 2018/2019

DEDICATIONS

HERE WE HAVE REACHED THE END OF THE JOURNEY IN THIS DREAM, TO START ANOTHER DREAM

I WISH MY SUCCESS

TO MY DEAR PARENTS, MY MOTHER WHO HAS BEEN WITH ME AND SUPPORTED ME ALL MY LIFE, AND MY FATHER WHO SACRIFICED ALL HIS LIFE TO SEE ME I BECAME WHAT I AM, THANK YOU VERY MUCH MY DEAR PARENTS.

TO MY BROTHER

TO THE WHOLE FAMILY

ALL MY FRIENDS

MANEL

I dedicate this work to…

All those who participated from near or far in the orientation of

my life and my mind.

Especially, to my mother,

To my father,

To my Islam, RafikandKrim

To my sisters Besma and Hanane

To my dear girlfriends Anissa ,Djouhaina ,Saliha and Imen

And all my friends.

DAOUD Rima

Acknowledgements

First, we would like to thank "ALLAH", our creator for giving us strength to accomplish this work.

Our sincerest gratitude to Dr. BOUKHAMLA Akram for accepting to supervise us, and for their incontestable instructions, patience, skills, and remarks that have us benefited.

Also our gratitude to all those teachers in the department of Computer Science, University KASDI MERBAH OUARGLA.

Also our colleagues in promotion Master 2019-2020

Abstract:

Domain Name System (DNS) is a primordial component of today's Internet for mapping Internet names to addresses and is required for nearly every network transaction. There are however, some security problems with DNS, due to the vulnerabilities such as "cache poisoning", "denial of service attacks", Man In the middle… etc.

The goal of our work consists to reinforce DNS security by implementing secured exchange keys between servers to protect internet from the attacks, hijack and the counter command DNS data. We made an application that provides the ability to validate authenticity and integrity of DNS messages in such a way that altering DNS information anywhere in the DNS system can be detected. A key exchange is made by asymmetric algorithm (Diffie-Hellman) encrypted (signature "SHA-1 with RSA").

Key Words: DNS, IP, hash, SHA1, Diffie-Hellman, asymmetric, signature, RSA.

Résumé:

Aujourd'hui le système de noms de domaine (DNS) est un composant primordial de l'Internet, qui permet de mapper les noms Internet avec les adresses nécessaires pour toutes les transactions réseau. Cependant, Il existe certains problèmes de sécurité avec DNS, à cause des vulnérabilités telles que "l'empoisonnement du cache", "attaques par déni de service", Man In the middle… etc.

L'objectif de notre travail consiste à renforcer la sécurité DNS en mettant en œuvre des clés d'échange sécurisées entre serveurs afin de protéger Internet contre les attaques, le piratage et les contre faits DNS data. Nous avons créé une application qui permet de valider l'authenticité et l'intégrité des messages DNS de manière à pouvoir détecter toute modification d'informations DNS n'importe où dans le système DNS.

L'échange de clé est effectué par algorithme asymétrique (Diffie-Hellman) crypté (signature "SHA-1 avec RSA").

Mots Clés: DNS, IP, hachage, SHA1, Diffie-Hellman, asymétrique, signature, RSA.

ملخص:

يعد نظام اسم المجال (DNS) مكونًا أساسيًا لإنترنت للوصول إلها. ومع ذلك هناك بعض مشكلات الأمان في DNS . نظرًا لوجود ثغرات أمنية مثل "تسمم ذاكرة التخزين المؤقت"، و "هجمات رفض الخدمة"، ورجل في الوسط... إلخ.

الهدف من عملنا هو تعزيز DNS من خلال التأكد من حماية معلومات DNS. لذلك صنعنا تطبيقًا يوفر إمكانية التحقق من صحة وصلاحية رسائل DNS بطريقة يمكن بها اكتشاف بيانات DNS في أي مكان من نظام DNS.

يتم تبادل مفتاح بواسطة خوارزمية غير متماثلة (Diffie-Hellman) المشفرة (توقيع "SHA-1 مع RSA. ")

الكلمات المفتاحية: DNS ، IP، التجزئة ، SHA1 ، Diffie-Hellman، غير المتماثلة ، التوقيع، RSA.

List of Abbreviations

FQDN	Fully Qualified Domain Name
CPE	Customer-premises equipment
DSA	Digital Security Assemblies
PKCS	Public-Key Cryptography Standards
MD5	Message Digest 5
SSL	*Secure* Socket Layer
IETF	Internet Engineering Task Force
RFC	Request for comments
RRSIG	Resource Record Signature
DS	Delegation Signer
NSEC	Next SECure
KSK	Key Signing Key
ZSK	Zone Signing Key
TTL	Time-To-Live
DS	Data *Security*
TLD	Top Level Domain
IANA	Internet Assigned Numbers Authority
ICANN	Internet Corporation for Assigned Names and Numbers
RZM	Root Zone Manager
TAR	tape archiver

List of figures

List of tables

Contents

Chapter 1 DNS

General Introduction

General Introduction

The Internet is a worldwide electronic network that enables many independent computer networks to connect together by using a common connection, called an Internet Protocol (IP). One of such systems on the Internet that works great in a friendly environment, but is vulnerable now, is the domain name system (DNS). The primary job of the DNS is to take care of the mapping between domain names (like www.example.com) and IP numbers (like 192.168.1.1). This system has proven to be highly scalable and the Internet would not have grown so big, so fast, without it. However, there are also some security problems with DNS. It has been shown that it is possible to corrupt the system with false data. Instead of having www.example.com, mapping to 192.168.1.1 an attacker could make it point to an arbitrary IP address under his control. To address these problems and other shortcomings of the DNS an addition has been proposed, called the secure domain name system (DNSSEC). DNSSEC uses cryptographically signed responses to authenticate the results; this detecting falsified data [1].DNSSEC provides data origin authentication and integrity by using digital signatures. Each zone creates public/private key pairs to sign its data. This brings us authentication and integrity checking. The private key is held by the administrator and kept secret while the public key is added to the zone in a resource record called a DNSKEY record [2].
Our objective is to validate and secure DNS query by using the cryptography signing for provide the origin data, authentication and integrity. Our work depends on ensuring information and authentication from the hackers, to validate and sing data, we use cryptography encryption and decryption methods through exchange keys (public/private).

Thesis organization

Including this general introduction, our thesis is divided into three chapter:

- The first chapter will focus on the basic of DNS. First, we will talk about the history of development of DNS, as well as the definition. Then we will present the hierarchy of the DNS. In addition the delegation and the resource records. After that, we will explain the work of DNS. Then we will present the vulnerabilities, which attacks the DNS data.
- The second chapter will present the concepts of the DNSSEC, as well as the definition and the resource records. Then we will present the DNSSEC mechanism and the DNSSEC keys.
- The third chapter we will realize the application by using the mechanisms and techniques (Java, NetBeans, cryptography…).

Finally, the thesis is concluded by a general conclusion where we summarized the main results obtained.

Chapter I: DNS

1.1 Introduction

The Domain Name System (DNS) is one of the most important components of Internet infrastructure. DNS translates names, such as www.cisco.com, into IP addresses, such as 192.168.40.0 (or the more extended IPv6 addresses), so that computers can communicate with each other. DNS makes using Internet applications, such as the World Wide Web, easy.

In this chapter, we will review the basic of DNS; we will then present the hierarchy of DNS, delegation and Resource record. Later, we will explain how a DNS queries does work. Then we will talk about DNS vulnerabilities. Finally, our chapter will be concluded by a conclusion.

1.2 The History of DNS

In 1970's there was the ARPANET's **HOSTS.TXT** file, HOSTS.TXT mapped every ARPANET host's name to its IP address it was easily implemented and understood. The file was maintained by the SRI Network Information Center (the "NIC"). The NIC released updated versions of the file twice a week.

Around 1980, the ARPANET consisted of hundreds of hosts. The ARPANET changed networking protocols from NCP to TCP/IP.LANs became popular, that any host on the same LAN could use the ARPANET. The Problems with HOSTS.TXT:traffic and load, Name collisions, Consistency. For solving the problems the ARPANET powers-that-were launched an investigation into replacement for HOSTS.TXT. the Goals:

- ➢ To solve the problems inherent in a monolithic host table system.
- ➢ Have a consistent naming structure.
- ➢ Create a generic solution that can be used for multiple purposes.

In November, 1983**, Paul Mockapetris** then of USC's Information Sciences Institute, designed the architecture of the new system, called the Domain Name System or **DNS.**

The initial DNS RFCs were released in 1984:

- ➢RFC 882, "Domain Names - Concepts and Facilities"
- ➢RFC 883, "Domain Names - Implementation and Specification"

The transition plan was initially released in November 1983; transition to be completed by May 1984 [3].

1.3 What is the Domain Name System?

The Domain Name System (DNS) is a distributed database system that provides hostname-to IP resource mapping (usually the IP address) and other information for computers on an internetwork. Any computer on the Internet can use a DNS server to locate any other computer on the Internet. DNS is made up of two distinct components, the hierarchy and the name service. The DNS hierarchy specifies the structure, naming conventions, and delegation of authority in the DNS service. The DNS name service provides the actual name-to-address mapping mechanism [4].

1.4 The Hierarchy of DNS

DNS uses a hierarchy to manage its distributed database system. The DNS hierarchy, also called **the domain name space**, use of hierarchy for scalability, decentralized administration of the name space. The DNS hierarchy is an inverted tree structure, much like directory. The DNS tree has a single domain at the top of the structure called **the root domain**. A period or dot (.) is the designation for the root domain. Below the root domain are the top-level domains that divide the DNS hierarchy into segments. The domain name space is further divided into subdomains representing individual organizations [4]. Figure 1 illustrates the DNS hierarchy.

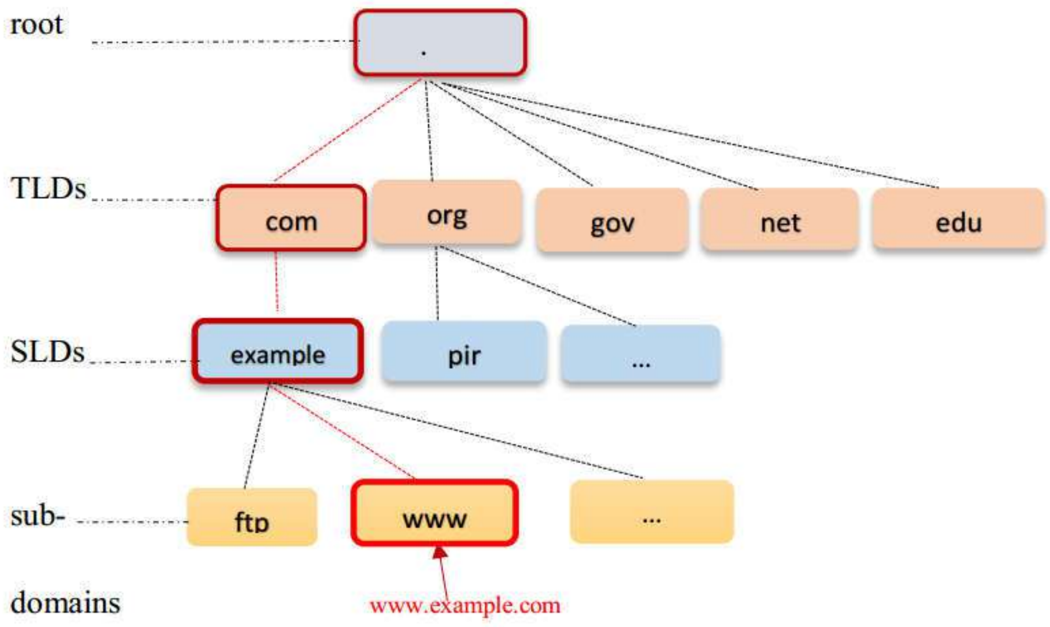


Figure 1.1 the Hierarchy of Domain Name System

1.4.1 Domain Names

The domain name represents an entity's position within the structure of the DNS hierarchy. A domain name is simply a list of all domains in the path from the local domain to the root. Each label in the domain name is delimited by a period Note that the domain names in the figure end in a period, representing the root domain. Domain names that end in a period for root are called fully qualified domain names (FQDNs). Each computer that uses DNS is given a DNS hostname that represents the computer's position within the DNS hierarchy [4].

1.4.1.1 Fully Qualified Domain Name (FQDN)

If a label is terminated by a null string, it is called a fully qualified domain name (FQDN). An FQDN is a domain name that contains the full name of a host. It contains all labels, from the most specific to the most general, that uniquely define the name of the host. For example, the domain name server1.cs.nec.edu. is the FQDN of a computer named server1 installed at the NEC Collete. A DNS server can only match an FQDN to an address. Note that the name must end with a null label, but because null means nothing, the label ends with a dot (.).

1.4.1.2 Top-Level Domains

The original top-level domains divided the Internet domain namespace organizationally into domains as show in table 1.1[5]

Table 1.1 Top-level DNS Domain

Domain	Used by
.com	Commercial Organizations
.edu	Educational Organizations
.gov	Governmental agencies
.mil	Military Organizations
.org	Nonprofit Organizations
.net	Networking entities
.int	International Organizations
.mil	Military
.aero	Air transport
.biz	Businesses
.coop	Cooperatives
.info	Informational
.museum	Museums
.name	People
.pro	Professionals
.cat	Catalan
.jobs	Employment
.mobi	Mobile devices
.tel	Contact details
.travel	Travel industry

Another top-level domain called **arpa** was originally used during the ARPAnet's transition from host tables to DNS. All ARPAnet's hosts originally had hostnames under **arpa**, so they were easy to find. Later, they moved into various subdomains of the organizational top-level domains [6].

Today, these original domains are called **generic top-level domains**, or gTLDs.

Table 1.2 Country code top-level domains

Domain	Used by
.us	United States of America
.uk	United Kingdom
.au	Australia
.de	Germany
.fi	Finland
.fr	France
.jp	Japan
.kr	South Korea
.nl	Netherlands
.se	Sweden
.dz	Algeria

The “generic” contrasts them with the country-code top-level domains, which are specific to a particular country.

1.4.1.3 Domains and subdomains

A domain is a label of the DNS tree. Each node on the DNS tree represents a domain. Domains under the top-level domains represent individual organizations or entities. These domains can be further divided into subdomains to ease administration of an organization's host computers. Any domain in a subtree is considered part of all domains above it.

Therefore, www.example.com is part of the example.com domain, and both are part of the .com domain [4].

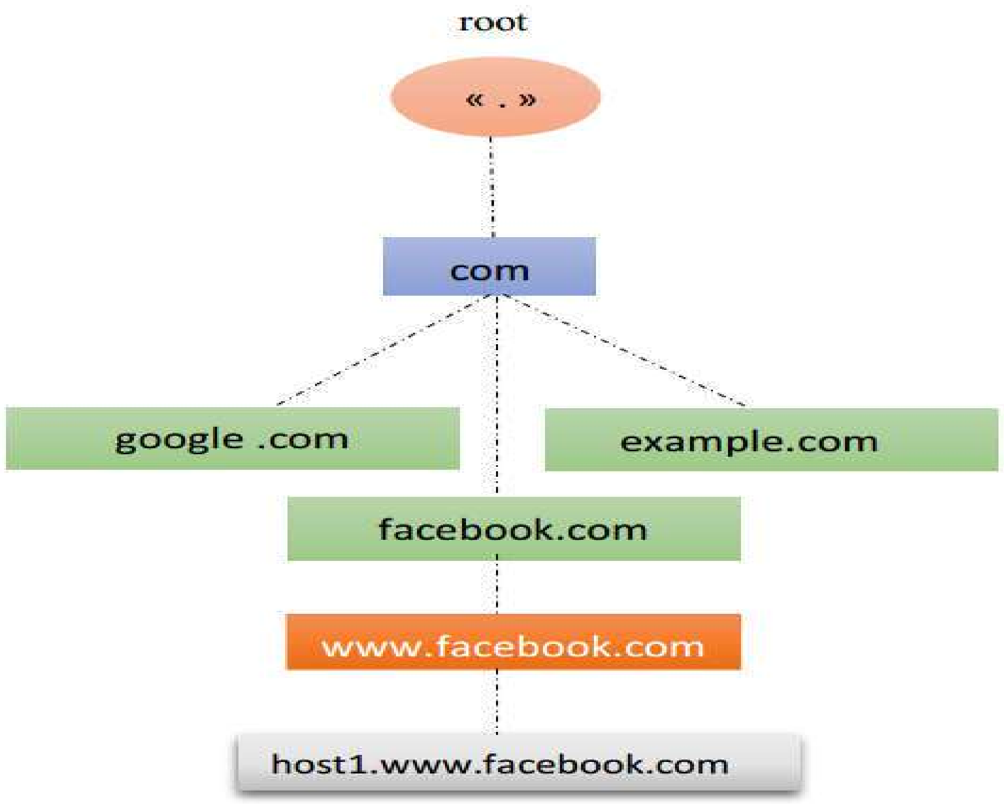


Figure1. 2 Domain and Subdomain

1.4.2 Delegation

Administrators often create subdomains to distribute management of the domain.

- An administrator can delegate responsibility for managing a subdomain to someone else.
- The parent domain retains pointers to the sources of data for the delegated subdomain.
- This sub-delegation provides for administrative scaling [6].

1.4.2.1 Delegation Creates Zones

Each time an administrator delegates a subdomain to someone else; this creates a new unit of administration.

- The subdomain and its parent domain can be administered independently.

- These units are called zones.
- The boundary between zones is a point of delegation in the name space [6].

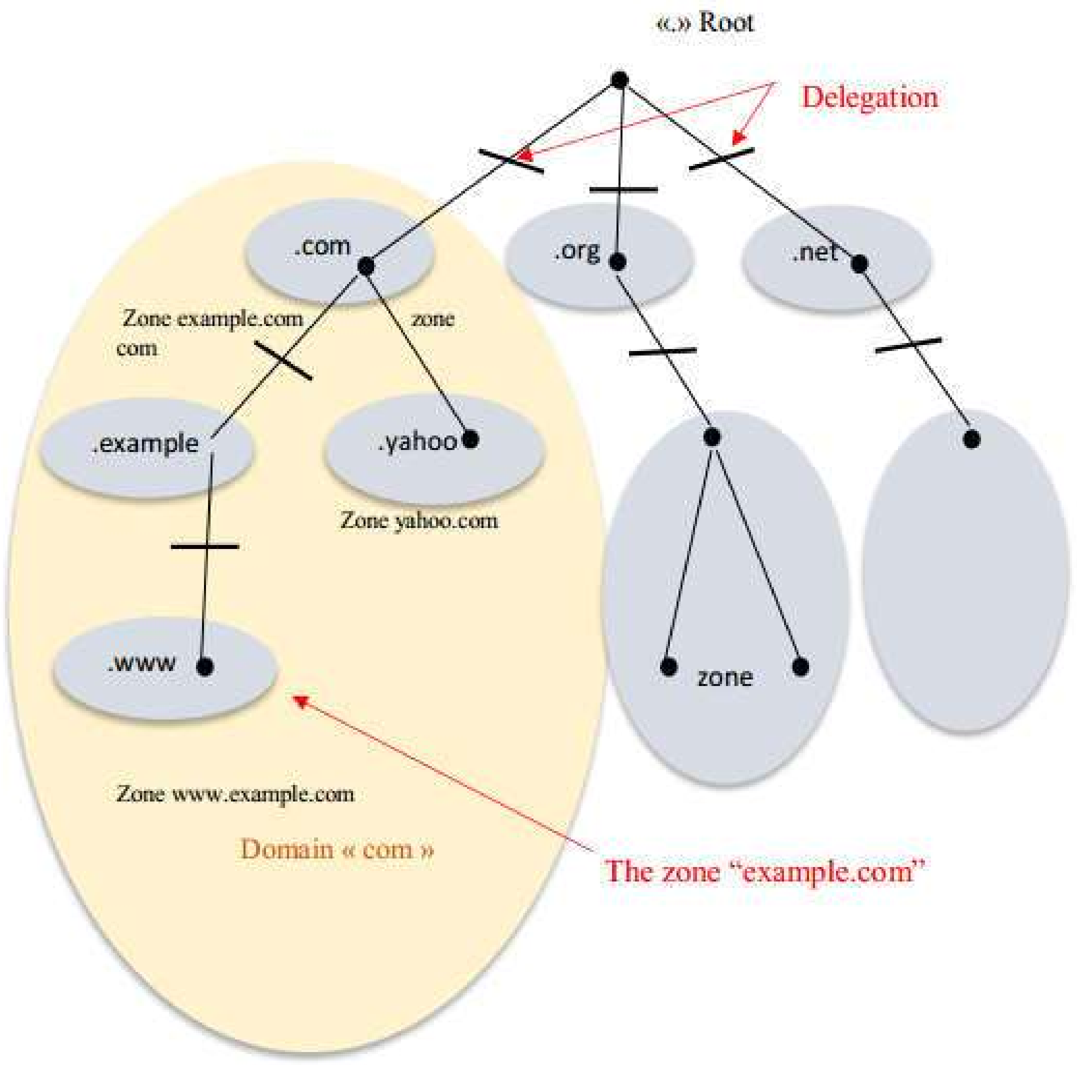


Figure 1.3 Delegation example

1.5 How Domain Name System works?

1. Client enters "www.example.com" where Client computer wants the IP address version of "example.com" and the first checks its own DNS cache for this information. It cannot find the IP address here if this is the first time using this website or the cache has been cleared [7].

2. The client computer request for the IP address of www.example.com is then redirected to the Internet Service Provider’s DNS Server. The ISP"s DNS server verifies its own cache. If the site has not been accessed before, it will be absent.
3. Every DNS server has a file that contains a list of all of the root DNS servers, at this the ISP"s DNS server redirects the query to the Root DNS Server [7].
4. The root DNS server maintains information about where a top-level DNS server (.com) is located and returns this information to the ISP"s DNS Server.
5. The ISP"s DNS server redirects the query to a top-level DNS server (.com).
6. The top-level (.com) DNS server knows the IP address of the DNS server for the “example.com” domain and returns that information to the ISP"s DNS server.
7. The ISP"s DNS server redirects the query to the actual DNS server for the “example.com” domain.
8. The DNS server for “www.exapmle.com” returns the IP address of the host of “www.example.com” to the ISP”s DNS server.
9. Lastly, the ISP"s DNS server sends the IP address to the client computer so the client can access “www.example.com” [7].

1.5.1 DNS Query Resolution

A name server can operate in two modes**, recursive** or **iterative**. A recursive query is sent with the **RD** (Recursion Desired) flag set to on in the DNS query header. In recursive mode, the name server searches through the DNS hierarchy in response to queries and returns either an error or the answer, but never referrals to other name servers. For iterative queries, the queried name server operating in iterative mode consults its own database for the requested data. If it cannot find the answer, it typically gives the IP address of the closest name server that might know the result. The client repeats the request, this time sending it to the server it just learned about. By default, queries to root name servers are iterative [8].

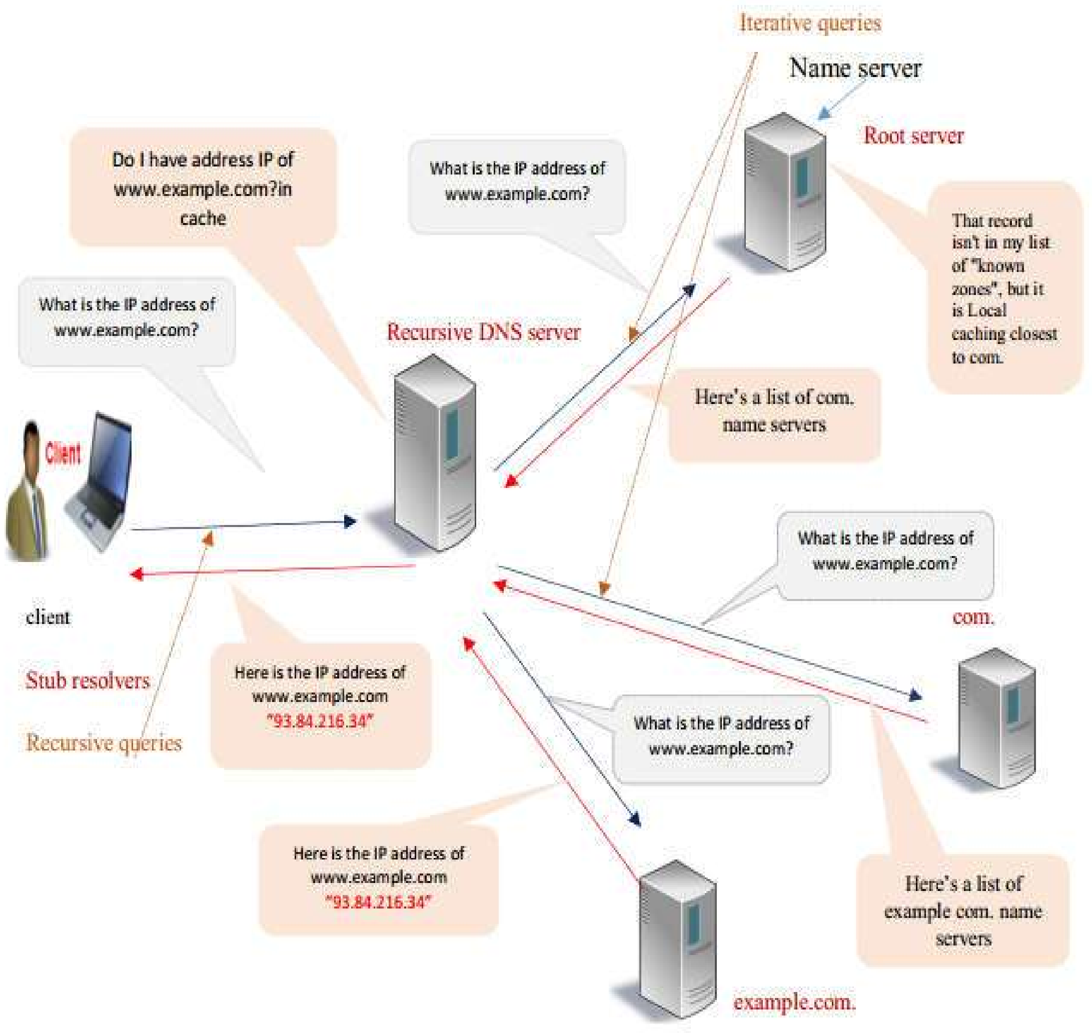

Iterative queries
Name server
Root server
That record isn't in my list of "known zones", but it is Local caching closest to com.
Do I have address IP of www.example.com?in cache
What is the IP address of www.example.com?
What is the IP address of www.example.com?
Recursive DNS server
Here's a list of com. name servers
Client
What is the IP address of www.example.com?
com.
client
Stub resolvers
Recursive queries
Here is the IP address of www.example.com "93.84.216.34"
What is the IP address of www.example.com?
Here is the IP address of www.example.com "93.84.216.34"
Here's a list of example.com. name servers
example.com.

Figure 1.4 A DNS Query Resolution

1.5.2 Name Servers

Are server programs, which maintain the information about the DNS tree structure and can set information. A name server may cache information about any part of the domain tree, but in general, it has complete information about a specific part of the DNS. This means the name server has authority for that subdomain of the namespace; therefore, it will be called **authoritative** [9].

For each zone, there must be a primary name server and a secondary name server

- The primary server (master server) maintains a zone file, which has information about the zone. Updates are made to the primary server.
- The secondary server copies data stored at the primary server.

1.5.3 Resolvers

Are programs that extract the information from name servers in response to client requests. It is assumed that the reader is familiar with the basic notions about DNS [9].

1.5.4 DNS server

Is also web server: it's primary objective is to interact with aforementioned database these DNS servers translate the domain name entered into the URL area of a web browser to the corresponding IP address these are thousands of DNS server worldwide which from the domain system which currently is the largest digital database.

1.5.5 Type of DNS server

There are two type of server: **authoritative** server and **recursive** server

1. **Authoritative server**

- Gives answers for specific zones.
- **"Authoritative"** for these zones.
- Only respond to queries for these zones.
- Never ask other DNS servers anything.
- A server can be authoritative for more one zone.
- A zone should have more one authoritative server [10].

2. **Recursive server**

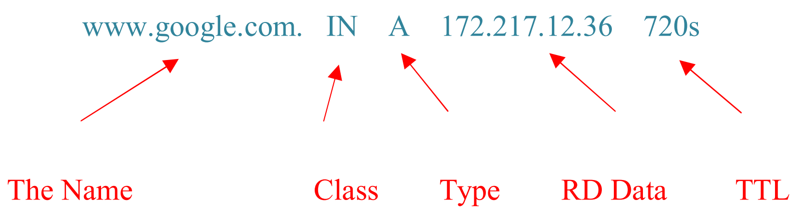
- Receives queries from clients
- CPE, user's PCs, mail servers, …etc.
- Send queries to authoritative servers
- Follow referrals down from the root [10].

1.6 Resource Records

A Resource Record (RR) is the basic data element in the domain name system. Resource Records define data types in the Domain Name System (DNS). The resource records are created to help DNS work like zone, so these are also called zone files. DNS records are used for mapping URLs to an IP [11]. These records are typically the connection of your website with the outside world. Requests for your website are forwarded to your DNS servers and then get pointed to the Web Servers that serve the website or to Email servers that handle the incoming email. Resource Records are stored in binary format internally for use by DNS software [12].However, resource records are sent across a network in text format while they perform zone transfers [13].Resource Records (RRs), have the format shown in figure 1.4.

Name	TTL	Class	Type	RD length	RD Data

Figure 1.5 Resource Record Format



The domain name is a pointer to the RR. The Time To Live (TTL) is the length of time that a cached RR may be considered to be valid. Class is the type of the network; RRs typically belong to the IN (Internet) class. The most commonly used values for the Type field [11]. Are listed in Table1.3. **RDLength** specifies the length of the RDATA field. Finally, RDATA describes the resource using the format specified by the type and class. An RRSet is a set of zero or more RRs, all of which have the same DNS name, class, and type [8].

Table 1.3: Resource Record Format

Type	Description
SOA	Start Of Authority
NS	Name server
A	IPV4 address
AAAA	IPV6 address
CNAME	Canonical Name(Alias)
MX	Mail Exchange

- **Start of Authority (SOA)**
  An SOA record marks the beginning of a zone file. It includes the name of the zone, its name server, a technical contact, a serial number, and timeout values.

```
google.com.  IN   SOA  server : ns1.google.com        22s  (00:00:22)
                       email  : dns-admin.google.com
                       serial  : 244646086
                        refresh : 900
                        retry : 900
                        expire  : 1800
                       minimum ttl : 60
```

serial number

- **Name server (NS)**
  Name server records identify the name servers that are authoritative for the zone, and they delegate subdomains to other organizations or zone files.

```
google.com.     IN    NS    ns1.google.com
```

- **A Record**
  ‘A’ records are the most common DNS record. They translate domain names like www.google.com into the IP addresses used by computers. A typical ‘A’ record would be:

```
www.google.com.   IN    A    172.217.12.36    720s     (00:04:30)
```

The IP addresses

- **AAAA Record**

  IPv6Address Record, rdata contains an IPv6 address.

```
www.google.com.   IN    AAAA  2607:f8b0:4000:811:2004  31s (00:00:31)
```

- **Canonical name (CNAME)**
  ‘CNAME’ (or canonical name) records are also common, but tend not to be used as often as ‘A’ records. Instead of translating a DNS name to an IP address, they translate to another DNS name that has to be translated again to resolve an IP address.

An example of a CNAME record would be:

```
website.google.com. IN  CNAME   www.google.com.
```

The host or domain   the canonical name (alias)

- **Mail eXchange (MX)**

  'MX' (or Mail eXchange) records are used to define the mail servers that are used by a domain. You can have one or more MX records with each pointing to a different mail server. The order in which mail will be delivered to those servers is determined by the priorityof the MX records. Priority is an additional element of the MX DNS record and a positive integer. The lower this number, the higher the priority assigned to that mail server.

  An example of two MX records for a domain would be:

```
google.com.   IN  MX  10 mail1.google.com.
google.com.   IN  MX  20 mail2.google.com.
```

The higher priority   host receiving email

- **Pointer (PTR)**

  Pointer records establish the reverse mapping from addresses to names. PTR records should exactly match the forward maps. The lines illustrate PTR records for DMZ addresses in the example's outside view, including the NAT address pool.

```
36.12. 217.172 .in-addr.arpa. IN PTR www.example.com.
```

1.7 DNS Data Flow

1. Resolver queries recursive server.
2. Recursive queries Master or Slave authoritative server.
3. Master server has the original zone data (zone file).
4. Slave server receives the zone data from Master [14].

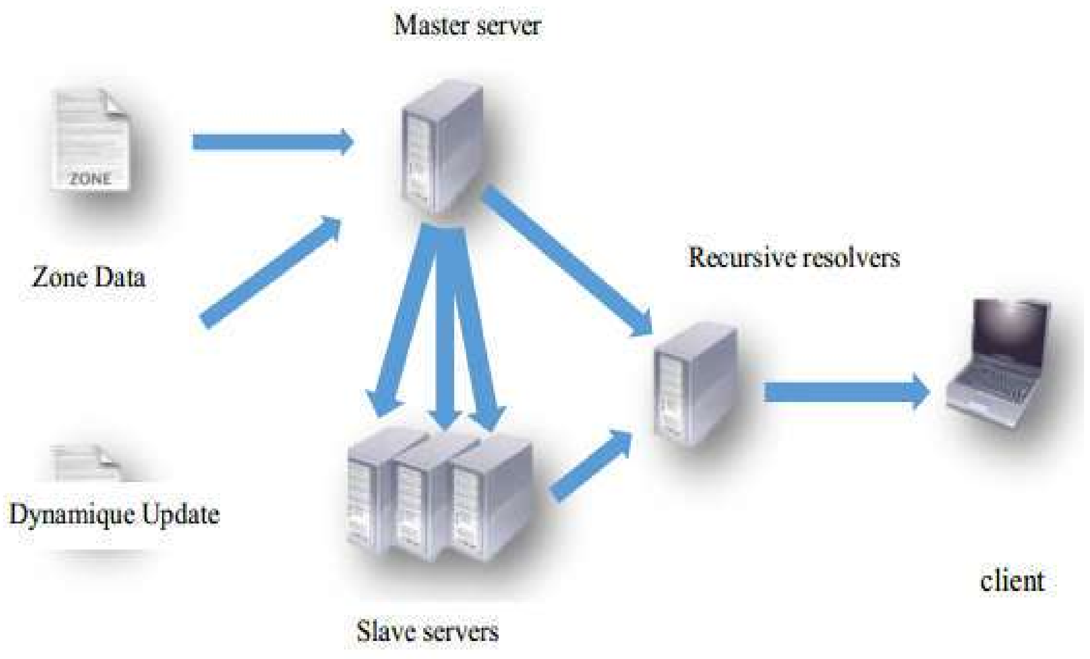


Figure 1.6 DNS Data Flow

1.8 DNS Vulnerabilities

We now briefly review some of the most important attacks on DNS:

1.8.1 Man in the middle (MITM) attacks

The recipient of data from a DNS name server has no way of authenticating its origin or verifying its integrity. This is because DNS does not specify a mechanism for servers to provide authentication details for the data they push down to clients. A resolver has no way to verify the authenticity and integrity of the data sent by name servers [8].

The Resolver can only authenticate the origin of a DNS reply data packet using the source IP address of the DNS server, destination and source port numbers and DNS transaction ID. An attacker can easily craft a DNS server response packet to match these parameters. The client has no choice but to trust as reliable the data provided by an attacker. An attacker can resolve legitimate queries, responding with false information.

1. 8.1.1 Packet Sniffing

DNS sends an entire query or response in a single unsigned, unencrypted UDP packet, which makes it easy to tamper with. By capturing DNS query packets, a wrong answer can be generated fast enough to reach the resolver before the correct answer from the name server. Compromising a router on a transit network allows an attacker to capture the DNS Reply packet from the name server and modify it. As no source authentication or data integrity checks are supported, this will not be detected by the resolver [8].

1. 8.1.2 Transaction ID Guessing

An attacker can respond with false answers to a predicted query, without having to be on the LAN to intercept packets. These answers will be cached either by the resolver or by the caching name server. The DNS Transaction ID field is only a 16-bit field, and the server UDP port associated with DNS is 53. On the client there are only 2^{32} possible combinations of ID (2^{16}) and client UDP ports (2^{16}) for a given client and server. In practice the client UDP port and the Transaction ID can be predicted from previous queries. It is common for the client port to be a known fixed value due to firewall restrictions, or the port number will increase incrementally due to resolver library behavior. The DNS transaction ID generated by a client usually increases incrementally. This reduces the search space to a range smaller than 2^{16} [15].By itself, ID guessing is not enough to allow an attacker to inject bogus data. This has to be combined with knowledge or guesses about Queries (QNAME) and Query type (QTYPE) for which a resolver might be querying. This can, for example, be achieved by cache snooping [16].

1. 8.2 Caching Problems

Through the use of caches, the DNS sacrifices consistency in favor of reduced access time. DNS caching raises concerns about cache inconsistency and staleness of data. Stale information may include security critical information, e.g. a compromised key. The current DNS protocol DNS resolvers use either recursive or iterative queries DNS does not support any means to propagate data updates or invalidations to DNS servers or caches in a fast and secure way [8].

1. 8.2.1 Cache Poisoning using Name Chaining

This attack introduces false information into DNS caches. This is achieved by means of DNS RRs whose RDATA portion includes a DNS name which can be used as a hook to let an attacker feed bad data into a victim's cache. The most affected types of RRs are CNAME, NS, and DNAME RRs. False data, associated with these names, can be injected into the victim's cache via the Additional section of the response. An Attacker can introduce arbitrary DNS names of the attacker's choosing, and provide further information that is claimed to be associated with those names [15].

1.8.2.2 Cache Poisoning using Transaction ID Prediction

In this attack, a large number of resolution requests are sent to the victim server (ns1.victim.com, say) with spoofed source IP addresses to resolve a name, say www.mybank.com. Each request will be assigned a unique transaction ID and processed independently. Since ns1.victim.com is trying to resolve each of these requests, the server will be awaiting a large number of replies from ns1.mybank.com. The attacker uses this wait stage to bombard ns1.victim.com with spoofed replies from ns1.mybank.com, stating that www.mybank.com points to an IP address which is under the attacker's control. Each spoofed reply has a different transaction ID, a source port and the spoofed DNS server IP address. The attacker hopes to guess the correct transaction ID and source port used by the querying name server. Once the attack is successful, false information will be stored in ns1.victim.com's cache [17].

1.8.3 DDoS attacks

DDoS attacks can have a significant impact on the global DNS database and its users. They are usually directed at root servers. This was evident with the recent DDoS attack in June 2004 [18], which was a repeat of a similar attack in October 2002[19]. These attacks caused a loss of availability of name resolution services to the Internet community.

1.8.4 Other significant DNS attacks

1.8.4.1 Information Leakage

A successful zone transfer by an attacker would serve as a reconnaissance attack, potentially revealing sensitive information about internal network configuration, e.g. the IP addresses of Internal firewall interfaces. DNS names could for example, represent project names that may be of interest to an attacker, or could reveal the identity of the operating system running on the machine [8].

1.8.4.2 DNS Dynamic Update Vulnerabilities

Protocols such as 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 [20]. The authentication for such updates is based solely on source IP address, and is vulnerable to threats such as IP spoofing. These attacks can range from denial of service, including deletion of records, to redirection [21].

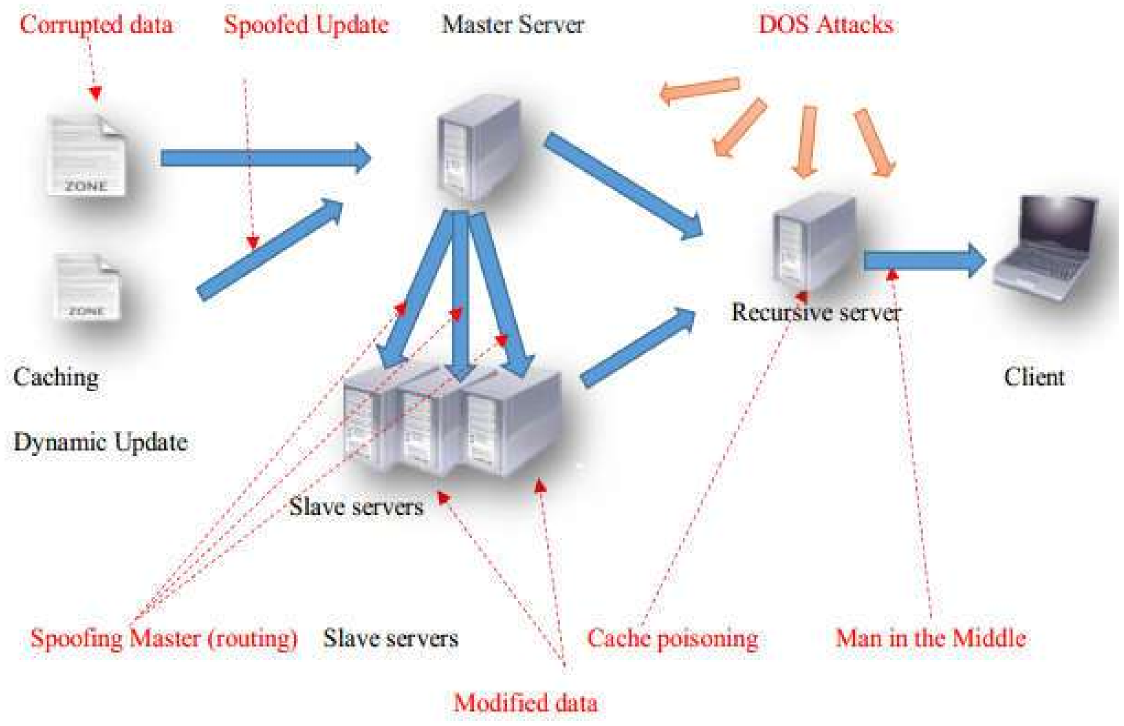


Figure1.7 DNS Vulnerabilities

1.9 Conclusion

In this chapter, we made an overview about DNS by talking about history of development of DNS, as well as the definition and the hierarchy of DNS. In addition, the resource records in a domain, which are stored in zones on DNS servers and finally we put problem statement about vulnerabilities in the DNS such as cache poisoning and traffic diversion that we will discuss the set of extensions, which is collectively known as DNS Security Extension (DNSSEC) for solving this problem in second chapter.

Chapter II: Presentation of DNSSEC

2.1 Introduction

The vulnerabilities in the DNS were discovered that allow an attacker to hijack the session and copy the authentication information of account. These vulnerabilities have increased interest in introducing a technology called **DNS Security extensions (DNSSEC). DNSSEC** is a cryptographic method for adding data origin authentication and data integrity verification to the zone data, which sets more stringent requirements for operational and technical control than standard DNS [7]. In this chapter, we will present in first time a small definition about the cryptography. In addition, the concept of DNSSEC will be presented, as well as the resource records of DNSSEC. Then we will show the DNSSEC mechanisms. After this, the keys of DNSSEC will be presented.

2.2 Cryptography

2.2.1 Definition

Cryptography is a field of computer science and mathematics that focusses on techniques for secure communication between two parties. This is based on methods like encryption, decryption, signing, generating of pseudo random numbers, etc [22].

2.2.2 Cryptographic Fundamentals

The four ground principles of cryptography are:

a. **Confidentiality** Defines a set of rules that limits access or adds restriction on certain information.
b. **Data Integrity** Takes care of the consistency and accuracy of data during its entire life cycle.
c. **Authentication** Confirms the truth of an attribute of a datum that is claimed to be true by some entity.
d. **Non-Repudiation** Ensures the inability of an author of a statement respective, a piece of information to deny it [22].

DNSSEC uses cryptography for two purposes: Integrity, Authenticity

2.2.3 Cryptography for DNS admins

To provide Authenticity and Integrity, we use Asymmetric Cryptography, DigitalSignatures [23].

2.2.3.1 Asymmetric Cryptography

Key pairs (Public and Private) Key Portions Data encrypted with one piece of a key can be decrypted or checked for integrity with the other. It is unlikely that a person holding the public key will be able to reverse engineer the private key [23].example for key: RSA, DSA, El Gamal, Diffie-Hellman, and PKCS.

RSA

RSA algorithm is asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. **Public Key** and **Private Key.** As the name describes that the Public Key is given to everyone and Private key is kept private.

An example of asymmetric cryptography:

1. A client (for example browser) sends its public key to the server and requests for some data.
2. The server encrypts the data-using client's public key and sends the encrypted data.
3. Client receives this data and decrypts it.

Since this is asymmetric, nobody else except browser can decrypt the data even if a third party has public key of browser.

2.2.3.2 Digital Signatures

If we combine hashes and public key encryption, we get a digital signature. We generate a hash, and then encrypt it with a key [23].

2.2.3.3 Hashes

Turns a string into a different series of characters and Fixed length.

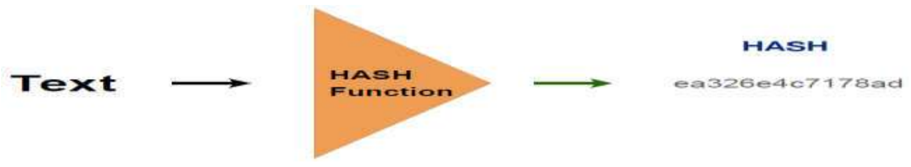


Figure 2.1 Hash Function

The most algorithms is used for hash are MD5, SHA.

- **SHA-1**

SHA-1 or Secure Hash Algorithm 1 is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value. This hash value is known as a message digest. This message digest is usually then rendered as a hexadecimal number, which is 40 digits long. It is a U.S. Federal Information Processing Standard and was designed by the United States National Security Agency.

SHA-1 is now considered insecure since 2005. Major tech giant's browsers like Microsoft, Google, Apple and Mozilla have stopped accepting SHA-1 SSL certificates by 2017.

To calculate cryptographic hashing value in Java, Message Digest Class is used, under the package java. Security.

2.2.4 Encryption Key

Key pair (private and public). Content encrypted with one key, can only be decrypted with the other one. A public key can "open" content encrypted with the private key, and vice versa .

2.3 History of DNSSEC

Table 2.1 History of dnssec[24].

1990	**Steven Bellovin discovers a major flaw in the DNS**
1995	Bellovin publishes his research; DNSSEC becomes a topic within IETF
1998	Dan Kaminsky discovers some security flaw
1999	RFC 2535, the DNSSEC protocol, is published: Three new
2005	RFCs published to update RFC2535 – RFC 4033 (DNS Security Introduction and Requirements) – RFC 4034 (Resource Records for DNS Security Extensions) – RFC 4035 (Protocol Modifications)
2005	In October, Sweden (.SE) becomes the first ccTLD to deploy DNSSEC
2008	new DNSSEC record created to address privacy concerns (RFC 5155)

2010	– In July 15, the root zone was signed – In July 29, .edu was signed – In December 9, .net was signed
2011	In March 31, **.com** was signed

2.4 Domain Name System Security extensions (DNSSEC)

DNSSEC protects Internet users from forged or manipulated DNS information, for example, what is known as DNS cache poisoning. Responses to DNS queries that are protected using DNSSEC are assigned a digital signature, the verification of which ensures that DNS information has not changed en route from the name server to the recipient system. DNSSEC, as the name implies, consists of a set of security extensions to the DNS protocol. DNSSEC introduces additional security related resource records with each reply, for the purpose of providing cryptographically signed integrity to the original DNS resource records. DNSSEC does not provide any confidentiality for DNS data does not guarantee delivery of resource records and does not provide integrity for unsigned portions of packets[25].The Domain Name System (DNS) security extensions provide origin authentication and integrity assurance services for DNS data, including mechanisms for authenticated denial of existence of DNS data."

DNSSEC is deployed on the three major components of the DNS infrastructure:

- **Recursive Server:** People use recursive servers to lookup external domain names such www.example.com. Operators of recursive servers need to enable DNSSEC validation. With validation enabled, recursive servers will carry out additional tasks on each DNS response they received to ensure its authenticity.
- **Authoritative Server:** People who publish DNS data on their name servers need to sign that data. This entails creating additional resource records, and publishing them to parent domains where necessary. With DNSSEC enabled, authoritative servers will respond to queries with additional DNS data, such as digital signatures and keys, in addition to the standard answers.
- **Application:** This component lives on every client machine, from web server to smart phones. This includes resolver libraries on different Operating Systems, and applications such as web browsers [26].

2.5 DNSSEC Resource Records

DNSSEC introduces four new resource record types:

- RRSIG (digital signature)
- DNSKEY (public key)
- DS (parent-child)
- NSEC (proof of nonexistence)

the DNSKEY, RRSIG, and DS records, which are used to build DNSSEC chains of trust, then we'll discuss the NSEC RR. DNSKEY RRs appear at a zone apex and store public keys used to authenticate that zone's data.

Each RRset (all RRs of the same name and type) in that zone is signed by the private key(s) corresponding to one or more of those DNSKEYs.

The digital signatures are stored in RRSIG RRs, which also include a time interval during with the RRSIG is considered valid.

- **DNSKEY** records, Contains Zone's public key(s) used in DNSSEC. Contains Zone's public key(s).Typically, each zone uses two DNSKEY records to sign DNS records, as discussed below.
- **RRSIG(Resource Record Signature)** records, which are cryptographic signatures of other records. Each RRSIG is a signature over all records of a given type for a certain name; this set is called an RRSet. For example, all A records for example.org will be authenticated by a single RRSIG (i.e., the example.org A RRSIG). Each RRSIG is created using the private key that matches a public key in DNSKEY records.
- **DS(Delegation Signer)** records, which are essentially hashes of child's DNSKEYs. The hash of the key is signed by the parent's DNSKEY and included in the parent's zone file, repeat for grandchild .which establishes the chain of trust reaching up to the root zone. The DS records in the parent zone are authenticated using RRSIGs, just like any other record type.
- **The NSEC (Next SECure)** RR is used to prove which data is and is not in a zone. Authenticated non-existence of data. One NSEC appears at each name in the zone, and it lists the RR types present at that name as well as the next name in a canonical ordering of the zone. A NSEC RR (along with the RRSIG(NSEC)) proves that there are no other records in the zone between the record which is the name of the NSEC and the 'next name' to which it points. The NSEC at the last name in that canonical ordering points back to the apex of the zone, forming a complete ring. Authoritative

servers generally send NSEC RRs as part of proofs that data does not exist (i.e. when an NXDOMAIN response would be appropriate). In particular, a signed NSEC is sent by each parent sending a delegation to an unsecured zone to prove that is has no DS RR for that child [27].

Table 2.2 Resource records added in DNSSEC.

DNSKEY	**Storing public key string**
RRSIG	Storing digital signature
NSEC	Storing next domain name
DS	Storing the hash of child's KSK-Public

2.6 Chain of Trust

A name server's public keys (KSK-Public and ZSK-Public) are shown in the DNSKEY resource records of its zone file. When a resolver wants to verify its received data, *D*, from a name server, it can perform a data integrity check. First, it gets the ZSK-Public key (as included in the DNSKEY record) from the server's response, and then uses the key to decrypt the signature (as included in the RRSIG record) to obtain the hash value of the original data, *H1*. Then, the resolver passes the received data, *D*, to the same hash function as the server use to calculate another hash, *H2*. After that, the resolver checks *H2* against *H1*. If they match, the received data, *D*, is verified and can be trusted. That is, the public key in the first step has not been verified and can be incorrect. Therefore, if an attacker could manage to compromise a DNS server, it could modify the zone data, and also generate a new key pair to re-sign the zone data again, then publish its public key to all DNS servers or clients. In this case, resolvers would not be aware of the problem, since they use the attacker's newly generated public key to verify its forged data, and hence, there would be no error during the authentication process. To solve the problem, chain of trust is used to build up an authenticated relationship between parent and child DNS servers. It requires that a parent server to verify the public key of its child server through the use of DS resource record[28].

2.7 DNSSEC keys

Each zone in DNSSEC typically has two public/private key pairs:

- **Key Signing Key (KSK)**: the KSK is a fundamental component of DNSSEC. Is used only to produce RRSIGs for DNSKEY records (hence the name).Only signs the Resource Record Set containing DNSKEYs for a zone. Used as the trust anchor. Needs to be specified in the parent zone using DS (Delegation Signature) records.
- **Zone Signing Key (ZSK)**: the ZSK is used to produce the RRSIGs for all other record types.Used to sign a zone, Can be lower strength than the KSK. No need to coordinate with parent zone if you want to change it.

2.7.1 Key Rollover

The recommendation is that key material for key signing keys (KSK) are only changed (rolled)
When needed, and that any changes of those keys are based on a well-balanced risk analysis.
Reasons for changing the KSK could include those personnel who have access to key material have left the organization or been given alternative work duties. Procedures for key rollovers should be designed relative to how key material for other systems (e.g., web servers, directory services and terminal services) are administered within the organization. Rolling the KSK poses a greater risk (compared to rolling the ZSK), as it would normally involve manual intervention, as it requires updating DS-records in the parent zone. Rolling key material for ZSK can however be handled automatically in most systems. Based on the relatively short key lengths, rollover is recommended every three months [29].
ZSK and KSK rollover schemes use different methods that share both similarities and differences.
ZSK rollover uses the Pre-publish method and the required steps are:

1. Generate new ZSK, add key to zone (Increase zone's serial number)
2. Re-sign zone with using old key and KSK
3. Wait for the TTL of the zone to pass
4. Re-sign with the new key but leave the old ZSK published in the zone
5. After all records signed with the old private key have expired (wait zone propagation time+ largest TTL of all records in the zone), remove old key
6. Resign one last time[2].

KSK rollover uses the Double Signature method and its steps are:

1. Generate new KSK, add new KSK to the zone and sign the DNSKEY RRset with bothkeys
2. Wait for the TTL of the zone to pass
3. Upload new DS to the parent zone
4. When new DS RR appears in the zone, wait TTL of the old DS record
5. Remove the old KSK and resign zone
6. Remove old DS record from parent [2].

2.8 DNSSEC mechanism

The following figure 2.2 show us the mechanism of the DNSSEC query and responses and how does it work, between the servers.

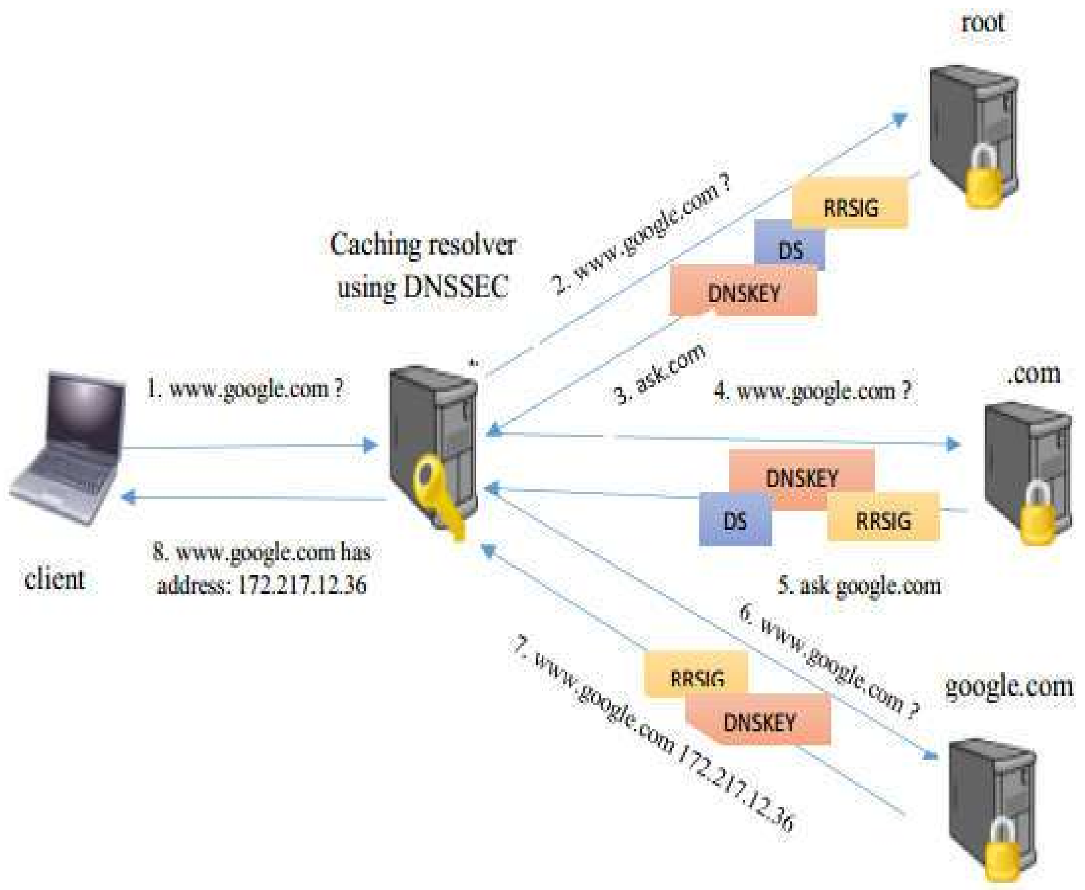


Figure 2.2 : Simple DNSSEC Query and Response

2.9 The Public Key DNS Security Extensions

PK DNSSEC uses three new Resource Records (*RR*) in order to provide end-to-end authenticity and data integrity: KEY (to encode the public key associated with a zone), SIG (to encode digital signatures over an *RR* set) and NXT (to indicate what does not exist in a zone). DNS servers are required to sign the *RR* sets in the zones for which they are authoritative, and answer queries by returning the corresponding SIG RRs along with the queried resource record set. An authenticated NXT RR is returned to indicate that a queried *RR* does not exist in the zone. On the other hand, a DNS resolver is required to verify signed answers by validating the SIG RRs that cover each *RR* set. The resolver can be configured to trust a set of public keys that correspond to a set of zones. If the answer is from a zone whose public key is trusted, the resolver can perform the verification without taking additional steps. Otherwise, the resolver needs to establish a chain of trust starting from one of the trusted public keys (usually of the root name server) down to the public key of that zone. During this process, the resolver may need to make additional queries for public keys of intermediate name servers [30].

2.10 Validating a DNSSEC record

The DNSSEC PKI is rooted at the KSK of the DNS root zone. This KSK is well known by DNSSEC-aware resolvers. Validating a DNS response starts at the root and continues down the DNS hierarchy: A resolver begins by using the KSK to validate the root DNSKEY RRSIG, which validates the root zone's ZSK. The resolver can then validate the child zone's DS record (and thereby the child zone's KSK) using the RRSIG for the DS records in the root zone, as this is signed with the root zone's ZSK. This process continues until the record in question is authenticated [29]. Figure 9 shows example records and how they are related.

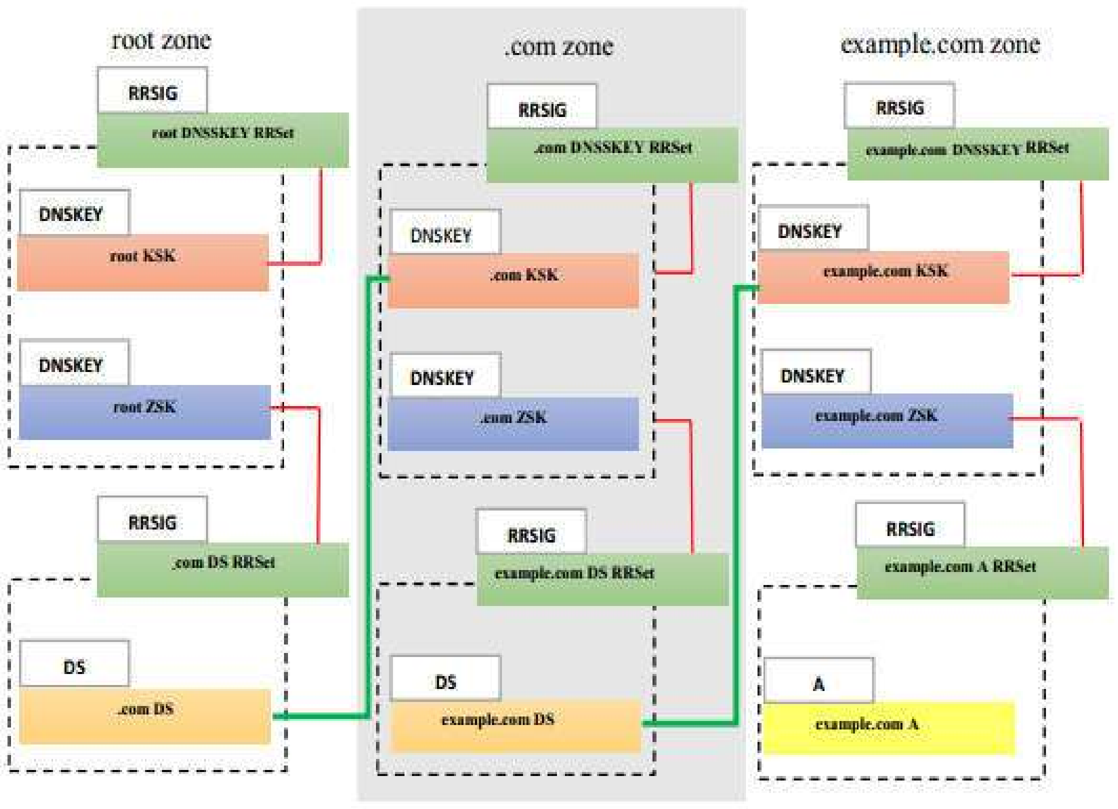


Figure 2.3 Overview of DNSSEC records necessary to validate example.com's A record. Each RRSIG is the signature of a record set (dashed lines) verified with a DNSKEY (redlines). Each DS record is the hash of a child zone's KSK (green lines).

2.11 Signing the root

The IANA root zone is both the authoritative list of TLDs and the starting point for delegation data to locate those TLDs. Without a single point (the root), the world would have to find TLD locations individually, without using the DNS and without any certainty that the TLDs were genuine. This clearly does not scale and is intensive and potentially error prone.

Although a single root is not a technical necessity, it is the Nominet view that, in practice, it is essential for the stability of the Internet. The DNS root database is managed by ICANN through the IANA function. IANA staff are responsible for updating the database of TLD managers. Currently, each update that is made to the DNS root database is reviewed by the US Department of Commerce prior to going live. This function has caused substantial debate at the international level, particularly during and after the World Summit on the Information

Society. Once the root database has been updated, the data that needs to change in the root zone is sent to the Root Zone Manager (RZM), currently VeriSign Inc., who then propagates this through the Internet to the other root server operators[31].

2.12 DNSSEC Deployment

Most DNS resolvers do not support DNSSEC validation, and many domains are not signed with DNSSEC. The fact thatthere is currently little support of DNSSEC further reduces amotivation for early adopters, since protection of DNSSEConly 'kicks in' when all the entities, involved in a resolutionof some domain, support DNSSEC.

1) **DNSSEC Validation at Resolvers**: A significant fractionof the resolvers currently signal DNSSEC support; however,less than 1% actually enforce DNSSEC validation. Obviously, for such resolvers, DNSSEC does not provide added security. This approach apparently assumes thatpermissive use of DNSSEC can provide evidence on whetherthe network can deploy DNSSEC fully without problems ornot, while notharming their security; however, in fact, suchresolvers are open to poisoning.
2) **DNSSEC Deployment at Zones*:***To make DNSSECvalidation effective in resolvers the domains have to adoptDNSSEC. However, most do not. DNSSEC adoption by top-level domains is not bad; the root and the largest domainssuch as com, net and orgsupport DNSSEC, and we found thatcurrently 30% of the TLDs use DNSSEC, apparently mostlythe larger.DNSSEC adoption is significantly worse for (important)second-level domains. We found that currently only 2% ofthe top 300,000 domains (according to Alexa) supportDNSSEC.Islands of security play an important role in the lack ofdeployment of DNSSEC. An island of security is a signedzone that does not have a DS RR at its parent, e.g., sincethe parent is not signed. Thus, the resolver cannot constructan authentication chain leading down from the trust anchor tothe target DNSSEC-enabled zone; as a result, resolvers cannotauthenticate the DNSKEY RR of the child zone.Currently, there are many islands of security; according to islands of security constitute 76.6% of the total numberof DNSSEC enabled zones on the Internet.To facilitate distribution of trust anchors of islands ofsecurity. The TAR maintains the public verification keysof islands of security. However, the TARs are not widelysupported since the zones are required to *add* their public keysto TARs and resolvers should be configured to query the TARto obtain the key when needed[32].

2.13 Disadvantages of DNSSEC

DNSSEC, like many things in this world, is not without its own problems. Below are a few challenges and disadvantages that DNSSEC faces [26].

- Increased, well, everything
- Different security considerations
- More complexity
- Increased fragility
- New maintenance tasks
- Not enough people are using it today

2.14 Conclusion

DNSSEC is the technical best practice in which the validity of DNS query is ensured through cryptographic signing. In this chapter, we have presented the basics of DNSSEC including the definition, as well as the resources records of DNSSEC and the chain of trust. In addition, the DNSSEC keys and DNSSEC mechanisms.

Chapter III: Experimental Results and Discussion

3.1 Introduction

In this chapter, we apply DNS security programming for the optimization of the transition the original DNS protocol. Of based on problems and disadvantages mentioned in Chapter II. For achieve this goal, we chose the application we'll be addressing.

3.2. Development environment

To realize this project we have used many techniques (language, Environment).
In the following sections we will provide different techniques.

3.2.1 The realization tools

- JAVA as programming language.

- NetBeans as an open-source integrated development environment (IDE) for developing with Java.

- We used **Cryptography** where:

- ➢ A way to encrypt or hash some content.
  - o Make it "secure" and/or verifiable.
- ➢ Intent is not always to hide the message.
  - o For DNSSEC, goal is to verify the content.
- ➢ Different methods and keys.

- Hashes:

- ➢ Turns a string into a different series of characters
- ➢ Fixed length

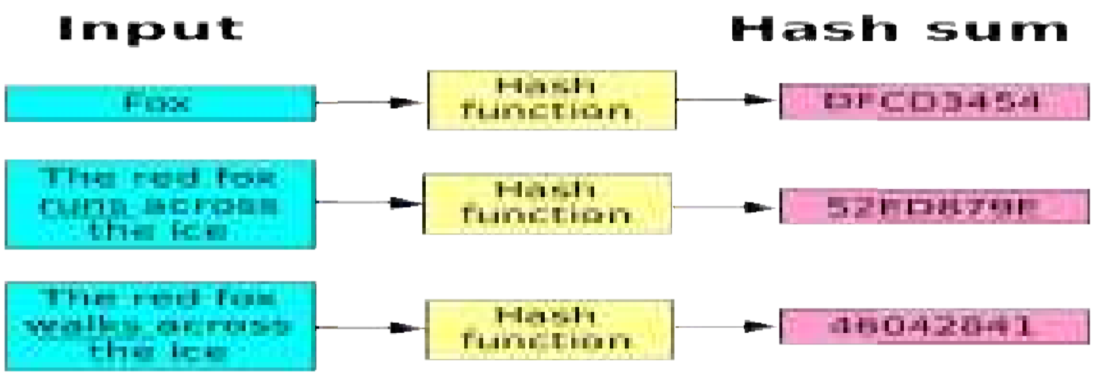


Figure 3.1 Hash function.

And exactly SHA-1, which takes an input and produces a 160-bit (20-byte) hash value known as a message digest – typically rendered as a hexadecimal number.

- Signature:

- Hashing + Encryption = Signature
- With private key.

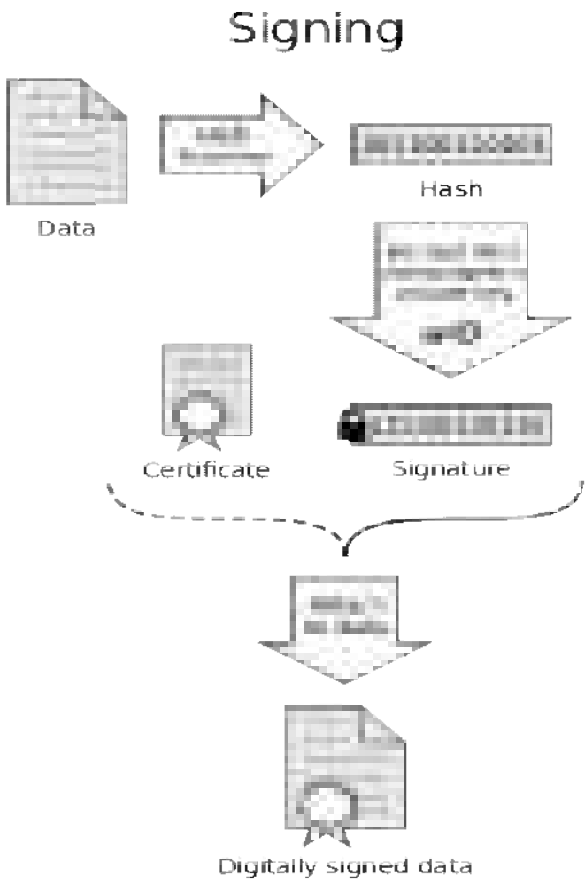


Figure 3.2: digital signatures.

And specifically, RSA algorithm asymmetric that it works on two different keys i.e. Public Key and Private Key.

- Verification:

- Encrypt in "RSA" the hash ("SHA-1") of the document with the public key.
- Compare this encryption with the signature of the document.

Or

- Decrypts in "RSA" the signature of the document with the public key.
- Compare this decryption with the hash ("SHA-1") of the document.

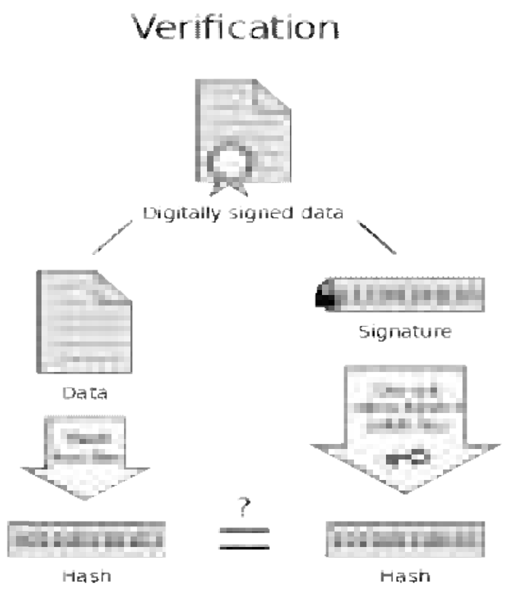


Figure 3.3: verifying the message with public key.

3.3 Experimental Results and Discussion

3.3.1 Application of DNS Security

- Image the whole file of Project.

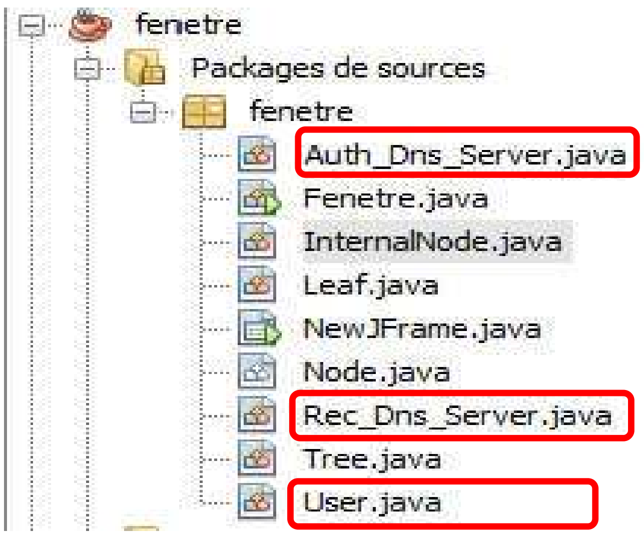


Figure 3.4: Image from Project.

There are three principal classes in our application Authoritative server (Auth_Dns_Server), Recursive server (Rec_Dns_Server) and browsers (User) shown in the **figure 3.4** taken from NetBeans.

3.3.1.1 Running of the application

- Method **User:**

In this class, the user is typing his query to be directed to the requested site address.

- Method **Rec_DNS_Server:**
  - Receives queries from users.
  - Send this query to Authoritative servers.
  - After the answer is sent by the Authoritative server, by Verify the content with public key and with public key.
- Method **Auth_DNS_Server:**
  - Receives queries from Recursive server.
  - Gives answers for specific zones.
  - Give the signature for that answers, hash (SHA-1) the answers and then it make encrypts (RSA) with private key.
  - Send to Recursive server
  - Application the DNS Security.

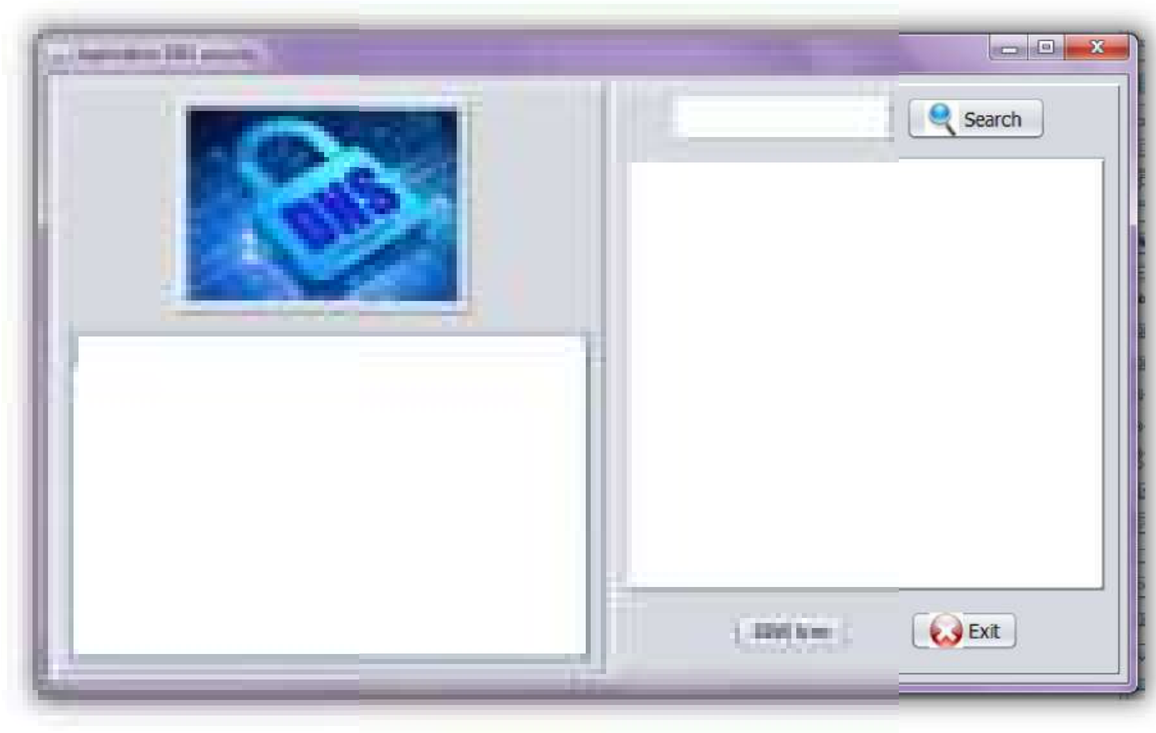
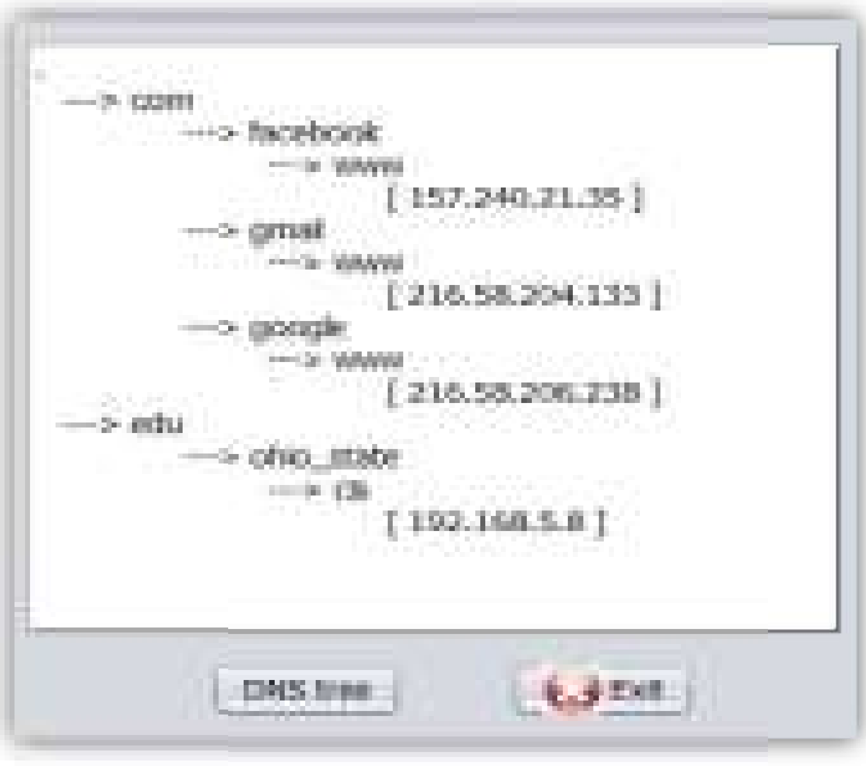


Figure 3.5 interface of our application.

The first interface we notice when running application of DNS security,with full contents to write our queries and work for DNS security.

3.4.1 DNS tree



```
---> com
          ---> facebook
                  ---> www
                          [ 157.240.21.35 ]
          ---> gmail
                  ---> www
                          [ 216.58.204.133 ]
          ---> google
                  ---> www
                          [ 216.58.206.238 ]
---> edu
          ---> ohio_state
                  ---> cis
                          [ 192.168.5.8 ]
```

Figure 3.6 The Hierarchy of DNS.

After pressing the button **DNS tree** shows DNS Hierarchy which we saved in data DNS. Domain Names hierarchical and each part of a domain name is referred to as either the root, top level, second level or as a sub-domain.

3.4.2 The results of the objective function

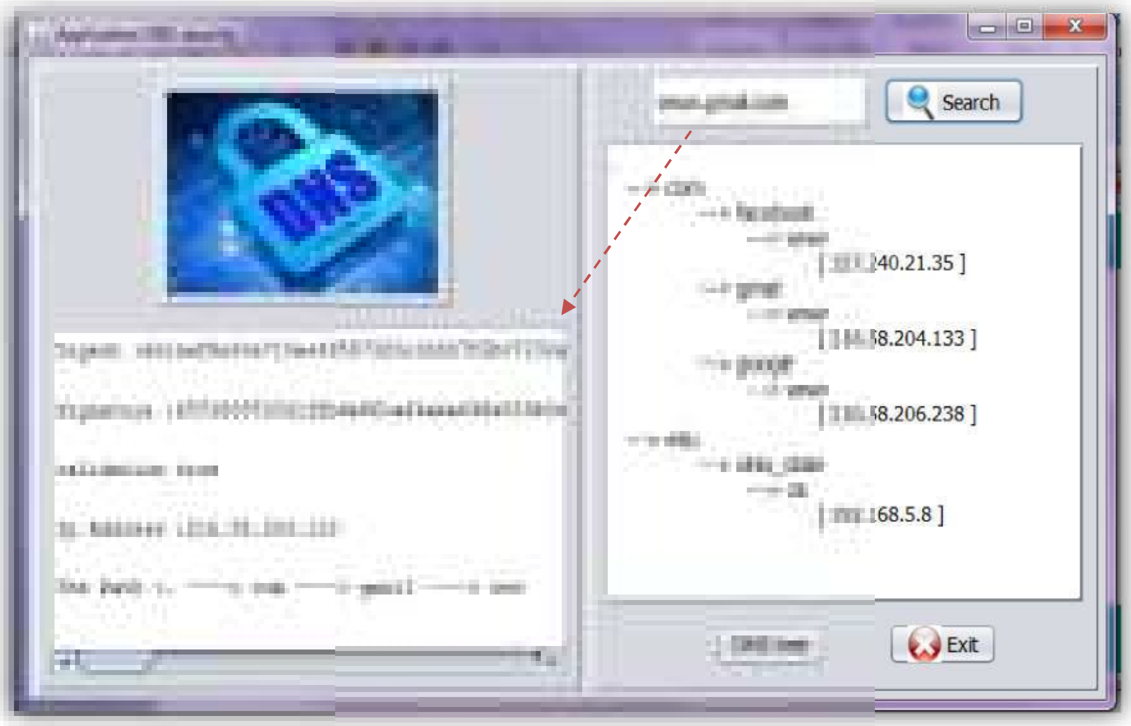


Figure 3.7 The first case is the correct request.

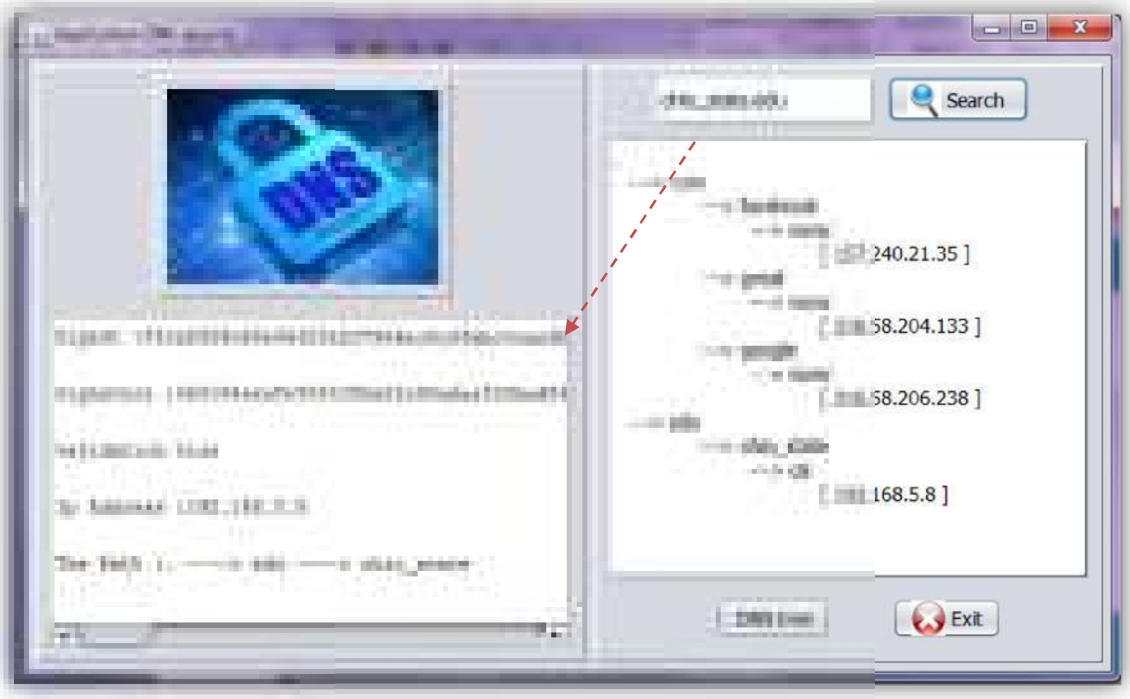


Figure 3.8 The second case is the correct request.

We noticed in the previous two pictures (**Figure 3.7** and**Figure 3.8)**, after typing a question from the user in the top right. In the left we see what happens before the answer, this is in case is correct request.

3.5 Conclusion

The security transmission of information in the network is more important in business and research, DNS servers communicate with each other to reach the desired site, and our application presents a get Chain of trust to validate DNS data.

We have used an algorithms and mechanisms to ensure the security and protection of Internet users.

We used public Key Encrypt it and Private Key of prove for server-to-server exchanges, and in the end of this chapter, we have shown the different parts of this work by some screenshots.

General conclusion

General conclusion

The Domain Name System has drawbacks from an operator's point of view. Where it has the problem of trusting the information that came from an incorrect reference, the name-based authentication process, and the problem of accepting additional information that was not requested and that may be incorrect.
Use two purposes Authenticity and Integrity, Cryptography is the art of protecting information by transforming it (encrypting it) into an unreadable format.
The goal of our work is to develop an application gets Chain of trust to validate data, to realize this project we followed these steps:

It was first necessary to define the concept DNS, where we mentioned the Introduction, history of DNS, definition of DNS, the Hierarchy of DNS, delegation, DNS works, resource records, DNS Data Flow, cite its Vulnerabilities.

In the second chapter we first touched the introduction About DNSSEC, definition and sections of the cryptography, history, Domain Name System Security extensions, DNSSEC Resource Records, Chain of Trust, DNSSEC keys, Key Rollover, DNSSEC mechanism, the Public Key DNS Security Extensions, validating a DNSSEC record, signing the root, DNSSEC Deployment and disadvantages of DNSSEC.

In last chapter we realized this project by discussing the scenario of DNS transactions between servers, experimental results and discussion, then the application the DNS security.
As a result, the DNSSEC is designed to guarantee the origin of DNS answers, and make hackers unable to replace the address by encryption and decrypt information between trusted servers. In addition, protecting users of websites and the site from stealing confidential information. If DNSSEC is deployed widely, users and applications are protected from the forged attacks.
In the future, we aspire to activate this feature on the servers.

References

References

[1] R. Gieben, "Chain of Trust: The parent-child and keyholder-keysignerrelations and their communication in DNSSEC", StichtingNLnet Labs.

[2] Anastasios Poulidis, HodaRohani, "DNSsec Revisited", In System and Network Engineering,University of Amsterdam, July 11,2014.

[3] David Conrad, "The Domain Name System", Nominum, Inc. David.Conrad@nominum.com.

[4] Novell, Inc., "Novell DNS/DHCP Services", 1800 South Novell Place Provo, Utah 84606 U.S.A,Management Utility Administration Guide, September 2, 2003.

[5] Cricket Liu, Paul Albitz, "DNS and BIND",**5th edition,** O'Reilly Media, Inc., the United States of America, **1992**.

[6] David Conrad "A Quick Introduction to the Domain Name System",Nominum, Inc., ITU ENUM Workshop,Jan 17, 2000.Auckland, New Zealand

[7] Sneha S. Shahane, Priyanka B. Shivgunde, Jyoti P. Dalvi, Madina M. Attar and Poonam V. Kabra,, "A Modified Approach for the Domain Name System Security" (DNSSEC) , International Journal of Engineering Research & Technology (IJERT) , Vol. 3 Issue 5, May – 2014.

[8]SuranjithAriyapperuma, Chris J. Mitchell "Security vulnerabilities in DNS and DNSSEC", Information Security GroupRoyal Holloway, University of LondonEgham, Surrey TW20 0EX, UK.

[9] Antonio Lioy, Fabio Maino, Marius Marian, Daniele Mazzocchi, **"**DNS Security", Italy, Terena Networking Conference, May 22-25, 2000.

[10] Mike Jager, **"**DNS Best Practices",Network Startup Resource Center, University Of OREGON.

[11] J.C.Klensin, J. Klensin "Role of the Domain Name System", February 2003

[12] J.C.Klensin, J. Klensin "Application Techniques for Cheking and Transformation of Names".

[13] Simar Preet Singh, “The Use of DNS Resource Records”, International Journal of Advances in Electrical and Electronics Engineering 230, No.-24, V1N2:230-236.

[14] Frederico Augusto de Carvalho Neves “DNSSEC” Registro.br May 17, 2010.

[15] D. Atkins and R. Austein. Threat analysis of the domain name system (DNS). RFC 3833, Internet EngineeringTask Force, Aug. 2004.

[16] L. Grangeia. Dns cache snooping. Technical report, Securing Team — Beyond Security, February 2004.

[17] Sainstitute. Attacking the DNS Protocol Security Paper v2. Technical report, Security Associates Institute, 2003.st

[18] J. Udell. Needed: Rapid internet response. Technical report, Infoworld ,www.infoworld.com, 2004.

[19] R. Naraine. Massive DDoS attack hit DNS root servers. Technical report, Internet News ,www.internetnews.com, 2002. 20

[20] P. Vixie, S. Thomson, Y. Rekhter, and J. Bound. Dynamic updates in the domain name system (DNS UPDATE). RFC 2136, Internet Engineering Task Force,Apr. 1997.18

[21] D. E. 3rd. Secure domain name system dynamic update. RFC 2137, Internet Engineering Task Force, Apr. 1997.

[22]Mohamed Barakat, Christian Eder, Timo Hanke,”An Introduction to Cryptography”,September 20,2018.

[23] Waterloo, “DNSSEC DomainNameSystemSecurityExtension”, 13 September 2016

[24] Michael Sinatra “DNSSEC: Signing, Validating, and Troubleshooting”, Energy Sciences Network Internet2/ESCC Joint Techs, Summer 2012.

[25] Matt Larson “DNSSEC Overview NANOG 51 Tutorial” 30 January 2010.

[26] “BIND DNSSEC Guide”, Internet Systems Consortium, Inc., 2017 .

[27] Taejoong Chung,Roland van Rijswijk-Deij,Balakrishnan Chandrasekaran, David Choffnes, Dave Levin,Bruce M. Maggs, Alan Mislove and Christo Wilson, “A Longitudinal, End-to-End View of the DNSSEC Ecosystem”, Open access to the Proceedings of the 26th USENIX Security Symposiumis sponsored by USENIX ,August 16–18, 2017

[28] Kin-Yeung Wong, Wei-Leung Koo, and Kai-Hau Yeung, “Enhancing the Security of Chain of Trustin DNSSEC” Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol II IMECS 2008, 19-21 March, 2008.

[29] Kirei AB, “ Recommendations for DNSSEC deployment at municipal administrations and similar organisations”, 2014.

[30] Reza Curtmola, Aniello Del Sorbo, Giuseppe Ateniese On the Performance and Analysis of DNS Security Extensions, Johns Hopkins University, Baltimore, MD 21218, USA.

[31] “Signing the root zone: A way forward toward operational readiness”, 15 July 2008 , https://www.icann.org/en/system/files/files/dnssec-paper-15jul08-en.pdf.

[32] Amir Herzberg and Haya Shulman, “Towards Adoption of DNSSEC: Availability and Security Challenges” ,Bar Ilan University.