



Security System for DNS Using Cryptography

Lalith Amudala¹, Sai Gopal Polsani², Ashwath Anand L A³, Vignesh M⁴ and Lavanya K⁵
^{1,2,3,4}Student, ⁵Assistant Professor(Senior), Department of Computer Science and Engineering,
VIT University, Vellore, Tamil Nadu 632014, INDIA

Abstract: The DNS generally means to verify whether the data comes from the authorized system. Weakness in security is DNS cache, DNS spoofing or due to weak authentication during the server exchanging updates. To provide the secure connection through or for the protocol that resolves hostnames and IP addresses since it is major concern to provide security and is also difficult. In this we use concept of digital signature and public key to encrypt the data. By generating a random Pseudo number, by compressing the message and signature is created using Private Key and transmitted along with the Public Key.

I. Introduction

The Domain Name System (DNS) converts the Internet domain and host names to IP addresses and vice versa. DNS converts the names we type in our Web browser address bar to the IP addresses of Web servers of sites. Many companies use DNS to manage their personal network. Networks at homes use DNS when accessing the Internet. DNS clients send requests and receive responses from DNS servers. Requests that contain a name, that result in an IP address being returned from the server, are called forward DNS lookups. Requests that contain an IP address and resulting in a name, called reverse DNS lookups. DNS has a database to store this name for all public hosts on the Internet. The DNS database resides on a hierarchy of special database servers. When Web browsers issue requests involving Internet host names, a piece of software called the DNS resolver first contacts a DNS server to determine the server's IP address. If the DNS server is unable to map, it will then forward the request to a different DNS server at the next higher level in the hierarchy. DNS also includes support for caching requests and redundancy. Most operating systems that are built to manage networks, support configuration of primary, secondary, and tertiary DNS servers, where each of these can service requests from clients. Domain Name System (DNS) services are at most times vulnerable, and DNS security has become a growing concern for many companies/organisations. At most times, DNS undergoes distributed denial of service (DDoS) attacks, and also reflection and amplification attacks. Because many companies/organisations use only a couple of DNS servers, DNS security can be easily bypassed by a volumetric attack, causing DNS servers to go offline and preventing users from accessing the website. So, DNS Security is very essential in our everyday lives as we must protect the data from attacks and eliminate all vulnerabilities. In this paper we are going to use cryptography (DES encryption) to implement the proposed model. In DES, the same key is used to encrypt and decrypt a message, so both the sender and the receiver should know and use the same private key. The DES is a block cipher, which means that a cryptographic key and algorithm are applied to a block of data one bit at a time rather than simultaneously. For a plaintext message to be encrypted, DES groups it into 64-bit blocks. Every block is en-ciphered using the secret key into a 64-bit cipher text using permutation and substitution. This process involves 16 rounds and can run in four various modes, by encrypting blocks individually or making every cipher block dependent on all their previous blocks. Decryption is simply the reverse of encryption, where the same steps are followed but reversing the order in which the keys are applied. The most basic method of attack for any cipher is brute force, which involves trying each key until you find the right one.

II. Literature Survey

The DNS security is designed to provide security by combining the concept of both the digital signature and the asymmetric (public) key cryptography. The public and the private key are used by the receiver and the sender respectively. To increase security to the DNS to face these security problems, the IETF included some security extensions to the DNS, which is known as DNSSEC [1]. The Domain Name System is a protocol that is used for mapping the domain names to protocols. Elliptic Curve Cryptography (ECC) is a public key. ECDSA is fast at verifying the signatures and uses small key size when compared to RSA and provides same level of security as given by RSA [2]. The analysis of network traffic data is considered for identifying largely agile patterns in DNS. If any change is observed, it is mapped onto the bipartite graph which is then checked for cyber crime [3]. The quantum key distribution (QKD) security relies on the no-cloning theorem, which allows not copying a quantum system properly. An efficient tool is proposed to study the quantum channel activity and arrange the properties of a quantum cloning-based attack for DV and CVQKD protocols [4]. Cryptography-based, prefix-preserving anonymization technique is developed that is provably as secure as the well-known TCP dpriv scheme,

and unlike TCPdriv, provides a consistent prefix-preservation in large scale distributed setting and evaluation of the security properties inherent in all prefix-preserving IP address anonymization schemes (including TCPdpriv) [5]. Caching is done in networks to make the data access in the internet easier. It is required very much for DNS. DNS has something called resource records which contains mapping of all names and addresses. When the reference cannot be done locally (client), the request is put to the server which then replies back to the client. All these caches have a time limit (TTL)[6]. The basic design of e-mail is vulnerable and can easily be attacked. The major threats are spam, phishing, denial of service (DoS). The prevention methods are SPF: The Sender Policy Framework (SPF) (Wong and Schlitt, 2006)· SDF: The Sender-ID Framework (SDF) (Lyon and Wong, 2006)· DKIM: The Domain Keys Identified Mail (DKIM) design (Allman et al., 2007; Lieba and Fenton, 2007)[7]. Nowadays vulnerability is found in almost all systems and hence system administrators must prioritize them according to the vulnerability. So, to prevent exploitation they set up ACL scripts to deny access to intruders [8]. Two key protocols are analysed state-aware and state-less protocols and also analyse security-goals and threats [9]. It analyses the threats that hamper cloud computing implementation on a large scale and also about the services offered by the current vendors. People by reading this paper would also get to know about the future of cloud security research and also know about the security risks associated with it [10]. The major reason for the business world's unwillingness to use the internet as a viable source of communication is the fear of security breaches. So they prefer cryptographic measures for developing internet's infrastructure over physical segregation. So they introduce IPSec architecture with security protocols in the network layer along with the transport layer security protocols [11]. And also to keep the vulnerabilities from exploitation of the hosts server and the network an improved method for network security which consists of the network management, the vulnerability scan, the risk assessment, the access control and the incident notification has been introduced [12]. And we also consider three email sender authentication mechanisms (i.e.) DNS : SPF, DKIM and Sender - ID Framework which are designed to assist in filtering of all undesirable mails in particular spam and phishing mails [13]. And to focus on the anonymizing the IP addresses in the trace, a cryptographic based prefix preserving anonymization technique is used and also the security properties of all prefix preserving anonymization schemes are evaluated [14]. Primarily taking into consideration the importance of the consumers in the online business world a secured, flexible and a cost effective cloud computing services are provided [15]. Optical cryptography based on computational ghost imaging (CGI) this is the major opportunity that converts plaintext into a random intensity vector it is a simplification of transformation it is vulnerable [16]. Evolving cloud technology which are provided by various web based services increased and data security is the major concern they explained security aware efficient distributed storage and efficient data distribution and conflation is explained [17]. Public key is generally preferred in the areas of data transformation to ensure data transfer is accessible in all the areas possible. BFW1 & BFW2 methods are explained for Public key transformation and Montgomery method to provide easy conventional product of integers for getting the converted data [18]. Prefix-preserving anonymization functions by mapping the set of functions to follow a canonical form at the same level of anonymity [19]. Most of the work is done through traditional cryptography to move towards initiation of hierarchical cryptography that works as the multiple mirror models with a strong baseless signature scheme is provided and is proven under Diffie-Hellman assumption [20].

III. Methodology:

DES ENCRYPTION:

The DES (Data Encryption Standard) algorithm is the most widely used encryption algorithm in the world. For many years, and among many people, "secret code is making" and DES have been synonymous. And despite the recent coup by the Electronic Frontier Foundation in creating a \$220,000 machine to crack DES-encrypted messages, DES will live on in government and banking for years to come through a life- extending version called "triple-DES."

DES works on bits, or binary numbers--the 0s and 1s common to digital computers. Each group of four bits makes up a hexadecimal, or base 16, number. Binary "0001" is equal to the hexadecimal number "1", binary "1000" is equal to the hexadecimal number "8", "1001" is equal to the hexadecimal number "9", "1010" is equal to the hexadecimal number "A", and "1111" is equal to the hexadecimal number "F".

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

For example, if we take the plaintext message "8787878787878787", and encrypt it with the DES key "0E329232EA6D0D73", we end up with the cipher text "0000000000000000". If the cipher text is decrypted with the same secret DES key "0E329232EA6D0D73", the result is the original plaintext "8787878787878787".

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

For example, take the message "Your lips are smoother than vaseline". This plaintext message is 38 bytes (76 hexadecimal digits) long. So this message must be padded with some extra bytes at the tail end for the encryption. Once the encrypted message has been decrypted, these extra bytes are thrown away. There are, of course, different padding schemes--different ways to add extra bytes. Here we will just add 0s at the end, so that the total message is a multiple of 8 bytes (or 16 hexadecimal digits, or 64 bits).

The plaintext message "Your lips are smoother than vaseline" is, in hexadecimal,

"596F7572206C6970 732061726520736D 6F6F746865722074 68616E2076617365 6C696E650D0A".

(Note here that the first 72 hexadecimal digits represent the English message, while "0D" is hexadecimal for Carriage Return, and "0A" is hexadecimal for Line Feed, showing that the message file has terminated.) We then pad this message with some 0s on the end, to get a total of 80 hexadecimal digits:

"596F7572206C6970 732061726520736D 6F6F746865722074 68616E2076617365 6C696E650D0A0000".

If we then encrypt this plaintext message 64 bits (16 hexadecimal digits) at a time, using the same DES key "0E329232EA6D0D73" as before, we get the cipher text:

"C0999FDDE378D7ED 727DA00BCA5A84EE 47F269A4D6438190 9DD52F78F5358499 828AC9B453E0E653".

This is the secret code that can be transmitted or stored. Decrypting the cipher text restores the original message "Your lips are smoother than vaseline".

Middle Ware:

Why middleware because DNS has its own functionalities to serve to provide data by making the query more optimal to search and to classify the query and its structure for optimal search results, the security is main concern making it to be a part of DNS may reduce the speed of transmission that the DNS concerns about. To avoid this we implement security as an external part of the system in Middleware as an hardware and software of the system or code working for security converting the data using the algorithm by DES encryption. Middleware adopts a end-to-end model of architecture for securing the data between consumer and DNS. It is transmission path of the packet these intercept the data by separating the components like data, header etc. When we talk about middleware we think about all the software components which allow the transfer of data through and hardware to satisfy its need to transfer. Considering the application to work on such as JAVA RMI and CORBA, from the middleware like the above CORBA is compatible with all other platforms and can take the input from any form like java, C, C++. But java is only compatible for its own program and COBRA is one of the most secure programming but most prefer JAVA RMI because it is easy to operate by getting from classes and CORBA supports only functions. Speed is also a constraint that transfer may suffer to its ability to work through security. Effective caching is the considerable task to overcome one of security issue in day to day dynamic working of internet and its failures and to sustain balanced load transmission and its distribution of data separation. This also improves reliability and secure operability of internet services. The techniques we apply are DES encryption cryptography. This operation is included in the middleware like in between the transfer of DNS and firmware of LAN or WAN. It is programmed with the suitable code such to support the hardware as proposed, to provide data authentication and integrity of data flow to the DNS. The system needs to keep the IP addresses in the frequent accessed list since the servers are used to access the DNS through frequent order because all the transmission is done through the servers that are in down line of transmission which depends on the service provider.

JAVA RMI:

The Remote Method Invocation(RMI) is an Application Program Interface(API) that helps us to create a distributed application in Java by providing some mechanism/process. The RMI accepts the request of objects to appeal methods on another object which is currently working/running in a different Java Virtual Machine(JVM). A remote communication is provided between the applications by the RMI using two objects namely skeleton and stub. These two objects are used by the RMI for communicating with the remote object which is processing in another JVM.

STUB:

Stub acts as a gateway to the client side i.e, the request which are sent are passed or routed through it. Hence it is at the client side. A stub does a few activities such as:

1. Provide a connection with the remote object in the JVM,
2. Transmit (marshals)the parameters that are written by it to the remote object (JVM),
3. Waits for the Acknowledgement (in the form of result),
4. The return value or the exception is read by it, and
5. Returns the value to the end user.

SKELETON:

Skeleton acts as a gateway for the server side i.e, all the requests that are coming from the client are routed through it. It does the following activities when it receives the request:

1. Read the incoming parameter for the remote method,
2. Invokes the method on the remote object, and
3. Writes and transmits(marshals) the result to the end user.

Example:

These are the 6 steps to write the RMI program.

1. Create the interface
2. Information of the interface is to be provided
3. Compile the implementation and create the stub and skeleton using the RMI compiler tool
4. Start the registry by RMI registry tool
5. Create and start the remote application
6. Create and start the client application

CORBA:

CORBA stands for Common Object Request Broker Architecture. It is a standard which is given by the Object Management Group(OMG) designed to make the communications of the system easier that are present on many different platforms. It is the world's leading middleware solution allowing the transmission of information to and fro, not depending on the hardware, operating systems and programming languages. CORBA is essentially a design for an Object Request Broker (ORB), where an ORB provides the process that is required by the distributed objects to communicate with one another, whether on the same or different devices, written in many different languages, or at different location on a single network.

The CORBA Interface Definition Language, or IDL, enables the development of location and language independent interfaces to different objects. Using CORBA, no matter where the application components are located they can communicate with each other. It is often described as a "software-bus" because it is highly based on the software dependent communication interfaces which help in locating and accessing of the objects.

Benefits of CORBA include:

1. Maturity
2. Open Standard
3. Wide platform support
4. Wide language support
5. Efficiency
6. Scalability
7. Data Encapsulation
8. Data Abstraction
9. Dynamic and Extensible solution

Distributed objects in any relationship have two side: the client and server. The main function of a server is to provide a remote interface and that of a client is to call a remote interface.

On the client side, there exists a remote object which has a stub method. This stub is wired into the ORB so that it calls the ORB's connection capabilities, which passes the invocation the called information to the server.

On the server side, the skeleton code is used by the ORB to translate the remote called information into a call on the resident object. The call and the parameters are translated by the skeleton to a certain format for implementation. When the result is returned, the errors are transformed by the skeleton code and send them back to the client.

Communication between the ORBs is done by using a shared protocol. IIOP, part of TCP/IP protocol, describes how CORBA IRBs pass information to one another.

JSON - JOSE:

Encryption using JSON (JavaScript Object Network) indicates that the data is encrypted using JSON based structures. The JSON Algorithms and IANA registers describe the cryptographic identifiers and algorithms for use in various specifications.

JOSE is a framework which is developed to provide an intended system with a method to transfer authorized data safe and secure between two different parties or groups. The JOSE mechanism's main purpose is to serve this method and hence provides a group of specifications to do so. To allow a system to apply access control own the resources present in it, JSON web token (JWT) consists some data. One of the main uses of the JWT is the authorization and authentication for the system.

Data is a set of keys that is useful for a target system as it provides the system with sufficient information about the client which helps it to set an access control to resources to an appropriate level. There are three classes of data namely: Public, Private and Registered (IANA). The public key is given to the users who want to access the information, the private key remains with the Administrator and this leads the users to access the data which can be accessed by them i.e, only to a level which the system specifies.

IV. DES ENCRYPTION: HOW IT WORKS

DES is a **block cipher**--meaning it operates on plaintext blocks of a given size (64-bits) and returns ciphertext blocks of the same size. Thus DES results in a **permutation** among the 2^{64} (read this as: "2 to the 64th power") possible arrangements of 64 bits, each of which may be either 0 or 1. Each block of 64 bits is divided into two blocks of 32 bits each, a left half block **L** and a right half **R**. (This division is only used in certain operations.)

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

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

L = 0000 0001 0010 0011 0100 0101 0110 0111

R = 1000 1001 1010 1011 1100 1101 1110 1111

The first bit of **M** is "0". The last bit is "1". We read from left to right.

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

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

The DES algorithm uses the following steps:

Algorithm:

1 Get the key from user

1.1 Get 64-bit key as input from the user. (to have correct parity, each byte should contain "1" bits in odd numbers. Every 8th bit is a parity bit.)

1.2 Calculate the key schedule.

1.2.1 Perform permutation on the 64-bit key. (The parity bits are removed, making the key to 56 bits. 1st bit of the permuted block is bit 57 of the original key, 2nd bit is bit 49, and 56th bit is bit 4 of the original key.)

1.2.2 Split the key after permutation into two halves. The first 28 bits can be called **C**[0] and the 28 bits from last are called **D**[0].

1.2.3 Calculate 16 subkeys starting with $i = 1$.

1.2.3.1 Perform either one or two circular left shifts on both **C**[$i-1$] and **D**[$i-1$] to get **C**[i] and **D**[i], respectively.

1.2.3.2 Do permutation for the concatenation **C**[i]**D**[i]. This will give **K**[i], which is 48 bits long.

1.2.3.3 Loop back to 1.2.3.1 until **K**[16] has been calculated.

2 Processing a 64-bit data block.

2.1 Get a 64-bit data block as input from the user. If the block is shorter than 64 bits, it should be considered as appropriate.

2.2 Perform permutation on the data block.

2.3 Split the block after permutation into two halves. The first 32 bits in the block are called **L**[0], and the last 32 bits are called **R**[0].

2.4 Apply the 16 subkeys that we found to the data block. Starting with $i = 1$.

2.4.1 Make the 32-bit **R**[$i-1$] into 48 bits by expanding.

2.4.2 Perform XOR for **E**(**R**[$i-1$]) with **K**[i].

2.4.3 Split **E**(**R**[$i-1$]) $\oplus$ **K**[i] into eight 6-bit blocks. Bits 1-6 are **B**[1], bits 7-12 are **B**[2], and so on with bits 43-48 as **B**[8].

2.4.4 Substitute those values in the S-boxes for all **B**[j]. Starting with $j = 1$. All the values in the S-boxes should be considered 4 bits wide.

2.4.4.1 Consider the 1st and 6th bits of **B**[j] together as a 2-bit value (let us call it m) indicating the row in **S**[j] to look for substitution.

2.4.4.2 Take the 2nd-5th bits of **B**[j] together as a 4-bit value (let us call it n) indicating the column in **S**[j] to find substitution.

2.4.4.3 Replace **B**[j] with value of **S**[j][m][n].

2.4.4.4 Loop back from 2.4.4.1 until all 8 blocks have been replaced.

2.4.5 Do permutation for the concatenation of **B**[1] through **B**[8]

2.4.6 XOR the resulting value with **L**[$i-1$]. Thus, your **R**[i] = **L**[$i-1$] $\oplus$ **P**(**S**[1](**B**[1])...**S**[8](**B**[8])), where **B**[j] is a 6-bit block of **E**(**R**[$i-1$]) $\oplus$ **K**[i]. (The function for **R**[i] is written as, **R**[i] = **L**[$i-1$] $\oplus$ **f**(**R**[$i-1$], **K**[i]).)

2.4.7 **L**[i] = **R**[$i-1$].

2.4.8 Loop back from 2.4.1 until **K**[16] has been applied.

2.5 Do permutation on the block **R**[16]**L**[16]

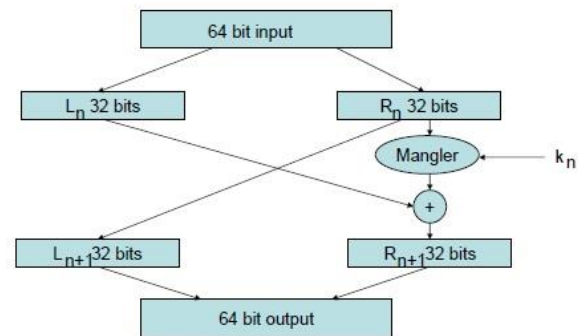


Fig.1: The above image depicts the working of DES Encryption.

To do a round DES Encryption/Decryption, Mangler function is used. The formula for the DES round is as follows

$$L_{n+1} = R_n, \text{ and}$$

$$R_{n+1} = L_n \oplus K_n(R_n)$$

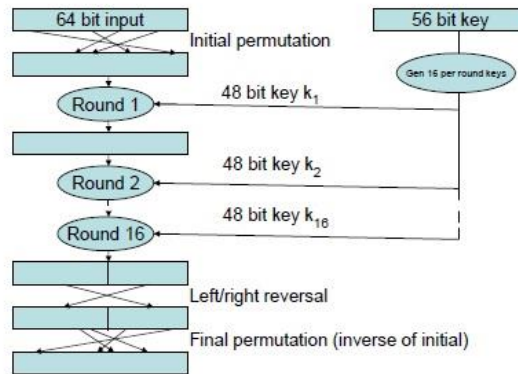
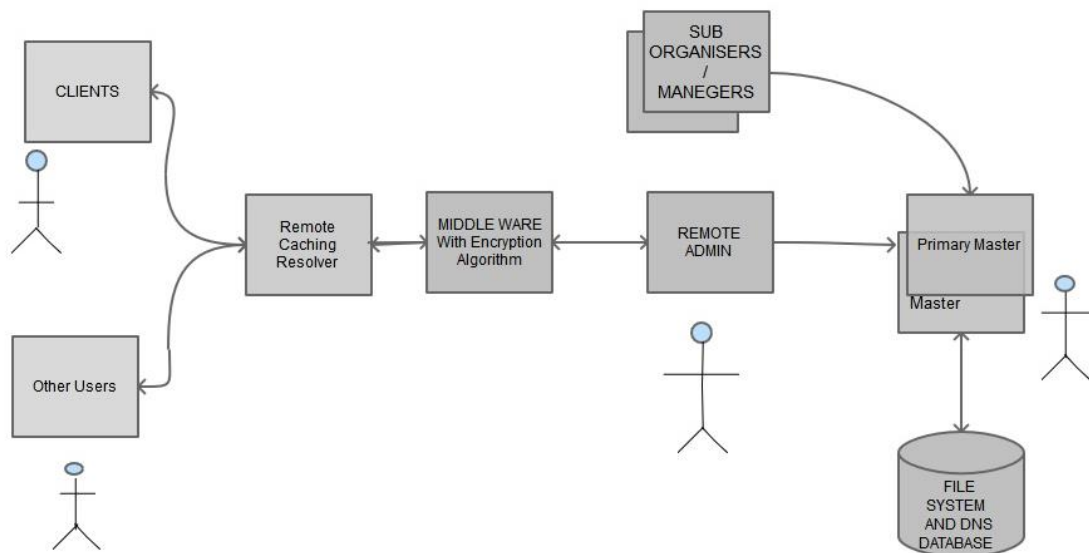


Fig. 2: The above figure depicts the DES round (encryption)

In DNS, this encryption is applied to secure the outgoing/incoming traffic from a client device to the DNS server.

2.6 The reverse of this algorithm performs decryption.



V. Conclusion

This explains the concerns in various contrasts that are faced in today's association of data redundancy and speed optimization. Security and speed are resolved from our explanation through problem statement with extra middleware included in the process which helps pressure on system which increases speed of which search algorithms work at its high potential and from which cost is reduced on DNS systems but ultimately increased on middleware components. We still intend to make improvements in the future by improving the speed and efficiency of the process so as to provide a faster user experience and also to combat emerging threats.

References

- [1] YEAR: 2013 Sachin Kumar Sinha, Avinash Kant Singh, Amaresh Sharma, B.TECH(IT), ITM Gida Gorakhpur, GBTU.
- [2] YEAR: 2015 Navin Kumar Tiwari M.Tech. (C.S.E), Sanjay Khakhil Assistant Professor Galgotia College of Engineering and Technology, Greater Noida.
- [3] YEAR: 2016 Mining agile DNS traffic using graph analysis for cybercrime detection Andreas Berger^a, Alessandro D'Alconzo^e, Wilfried N. Gansterer^b, Antonio Pescapé.
- [4] YEAR: 2014 Geometrical analysis of physically allowed quantum cloning transformations for quantum cryptography, Laszlo Gyongyosia, Sandor Imrea.
- [5] YEAR: 2004 Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme Jinliang Fana, Jun Xua, Mostafa H. Ammara, Sue B. Moonb.
- [6] Performance models for hierarchy of caches: Application to modern DNS caches by Sara Alouf, Nicaise Choungmo Fofack, Nedko Nedkov.
- [7] DNS-based email sender authentication mechanisms: A critical review by Amir Herzberg
- [8] Using the vulnerability information of computer systems to improve the network security by Yeu-Pong Lai, Po-Lun Hsia
- [9] Security concerns and countermeasures in network coding based communication systems: A survey by Vahid Nazari Talooki, Riccardo Bassoli, Daniel E. Lucani, Jonathan Rodriguez, Frank H.P. Fitzek, Hugo Marques, Rahim Tafazolli.

- [10] Cloud security: Emerging threats and current solutions by Luigi Coppolino , Salvatore D'Antonio , Giovanni Mazzeo , Luigi Romano
- [11] Refik Molva 23 April 1999, Internet security architecture.
- [12] Yeu-Pong Lai¹ , Po-Lun Hsia²20 September 2006, Using the vulnerability information of computer systems to improve the network security.
- [13] Amir Herzberg 21 January 2009 DNS-based email sender authentication mechanisms: A critical review.
- [14] Jinliang Fan^{a,1}, Jun Xu^a, Mostafa H. Ammar^a, Sue B. Moon^{b,1,2} 27 June 2003 Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme.
- [15] Saurabh Singh^a, Young-Sik Jeong^b, Jong Hyuk Park^a12 April 2016 A survey on cloud computing security: Issues, threats, and solutions
- [16] Sheng Yuan^{a,1}, Jianbin Yao^a, Xuemei Liu^a, Xin Zhou^b, Zhongyang Li^a Cryptanalysis and security enhancement of optical cryptography based on computational ghost imaging.
- [17] Yibin Li^a, Keke Gai^b, Longfei Qiu^c, Meikang Qiu^{1, b}, Hui Zhao (2015) Intelligent cryptography approach for secure distributed big data storage in cloud computing.
- [18] Satyanarayana Vollala, Ramasubramanian N(2015). Energy efficient modular exponentiation for public-key cryptography based on bit forwarding techniques.
- [19] Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme.
- [20] Lei Zhang^{a,1}, Qianhong Wu^b, Josep Domingo-Ferrer^c, Bo Qin^d, Peng Zeng^a (2015). Signatures in hierarchical certificate less cryptography: Efficient constructions and provable security.