Proposing a new version of Kerberos Authentication Protocol using ECC and Threshold Cryptography for cloud security

Thesis Report

SCHOOL OF ENGINEERING AND COMPUTER SCIENCE
Department of Computer Science and Engineering BRAC University

Supervisor: Hossain Arif

Mukti Rani Sutradhar (16101180)
Nigar Sultana (13301043)
Himel Dey(14101101)

Submitted on December 24, 2017

Certificate of Approval

Declaration: This is to certify that this thesis report is submitted by Mukti Rani Sutradhar (ID:16101180), Nigar Sultana (ID:12101009) and Himel Dey(ID: 14101101) for the degree of Bachelor of Science in Computer Science and Engineering to the Department of Computer Science and Engineering, School of Engineering and Computer Science, BRAC University. We hereby declare that this thesis is based on the results found by ourselves and the materials of work found by other researchers are mentioned by reference. The contents of this thesis, neither in whole nor in part have been previously submitted to any other Institute or University for any degree.

Hossain Arif
Assistant Professor
Dept. of Computer Science and Engineering
BRAC University Dhaka, Bangladesh

Signature of Author
Mukti Rani Sutradhar

Signature of Author
Nigar Sultana

Signature of Author
Himel Dey

Acknowledgement

First of all we want to thank our almighty to give us strength to finish the thesis. We are very much grateful to our supervisor **Hossain Arif,** Assistant Professor of Computer Science and Engineering, BRAC University for guiding us developing this system. It would have been really difficult to bring this work towards a completion without his guidance, enormous encouragement and continuous support. The period of thesis work is from January 2017 to December 2017. This project work is submitted to the Department of Computer Science and Engineering, School of Engineering and Computer Science, BRAC University in partial fulfillment for the requirement of Degree in Bachelor of Science in Computer Science. We declare that this work has not been submitted anywhere else for the award of any other degree.

Abstract

Nowadays cloud computing has become inevitable in every spheres of our work, that means not only in business sectors but also in nonbusiness related task. As it has many beneficial aspects we are relying on cloud computing day by day .The more we are depending on cloud computing, the more security issues are rising. Although there are some existing protocol which are serving security for users information in cloud, at present they are not secure enough to protect users' information in cloud. Kerberos authentication protocol is one of them which was once highly secured to authenticate data but with the passage of time and with the advancement of technology Kerberos authentication protocol is no longer that much secure. To improve its performances, many authors have proposed different kinds of model with a very common and popular public key cryptography called RSA cryptography. Of course RSA public key cryptography is good but it has some flaws which make this cryptography less efficient. As a result in our research we are trying to bring a new approach of using cloud more securely. Here, we are implementing Kerberos authentication process with Elliptic Curve Cryptography (ECC) as well as Threshold Cryptography to provide secure transaction of data which will be hard to break by third party or intruder, increase memory efficiency, cost efficiency as well as reduce the burden of computation. It acts as a third party between cloud server and clients to allow authorized and secure access to cloud services. We have also shown that why ECC public key cryptography is more appropriate than RSA public key cryptography.

Table of Contents

List of figures:

Chapter 1

Introduction

1.1 Introduction

Data and applications are maintained by cloud computing using the internet and central remote servers. With the help of cloud computing, we can install and access our personal files with internet access. use applications without installation and access our personal files on any computer but in this case, we need only internet access. By centralizing data storage, processing and bandwidth this technology allows for much more efficient Computing. Cloud storage is used by a large number of individuals and enterprises. All data stored on their hard drive is not cured by the user and no one knows where exact data saved. The security in the cloud is one of the most important issues. Already many researchers researched on cloud security problem. But we add another service to enhance the security issues. This service is Kerberos authentication service with threshold cryptography and ECC. In this theory, admin defines any IP address for using the cloud service provider. It means admin makes restrict for some user. At the next step the user with that IP address can connect to the Kerberos and after this service, it should connect to the cloud service provider for sending the data. So with this long filtering, we can enhance the security problem in the cloud. We will discuss the whole process of the security model. Although the Kerberos is alone not a perfect model for authentication because the single Kerberos server is prone to failure, so in this paper we are proposing two Cryptography algorithms known as a Threshold cryptography algorithm and ECC for increasing the security and availability of data. In threshold public key encryption system the private key is distributed among n decryption servers and in this case at least k servers are needed for decryption. The key can be retrieved by performing computation on at-least k no. of participants. Less than k no of participants are not useful to retrieve the key. Such a scheme is called a (k, n) threshold scheme. It is easily computable when having the necessary data available and Elliptic curve cryptographic schemes are public-key mechanisms that provide encryption, digital signature and key exchange capabilities.

1.2 Keywords: Cloud computing, Kerberos, Authentication, Protocol, Threshold, ECC, RSA

Chapter 2

2.1 Literature Review

This section provides a review of literature to set a foundation of discussing various cloud computing server security aspects. Bharill, Lalwani and Hamsapriya tried to secure the cloud service by using mathematical derivation of polynomial equation of Threshold and proposed a new version of Authentication Protocol. By filtering the unauthorized access and to reduce the burden of computation and memory usage of cloud provider this model can also be benefited against authentication checks for each and every user. It acts as a third party between cloud server and clients to enable approved and secure access to cloud services. [1]Amara and Siad talked about the possibility of Elliptic Curve Cryptography (ECC), and how it's a superior guarantee for a quicker and more secure strategy for encryption in contrast with the present standards in the Public-Key Cryptographic algorithms of RSA. They also said that Elliptic Curve Cryptography covers relevant asymmetric cryptographic primitives like advanced marks and key understanding algorithms. The function utilized for this intention is the scalar multiplication k.P which is the center operation of ECCs. Here, k is a whole number and P is a point on an elliptic curve. This paper clarifies the part of ECC in the system security.[2]Mehdi and K.venkat implemented Kerberos authentication process to secure cloud data storage and manage the user’s data in the cloud. The objective of their article is to apply on cloud data storage security and to manage the user's data in the cloud by Implementation of kerberos authentication service. they trust that this novel article is the foundation for the next opportunity of rising the cloud security. [3] Takabi, Joshi and Ahn discussed about unique issues of cloud computing that exacerbate security and privacy challenges in clouds and they also provide the adequate information about the structural view of cloud computing which are SaaS (Software as a Service) ,PaaS (Platform as a Services) and IaaS (Infrastructure as a service).[4]Wentao introduces some cloud computing systems and evaluates cloud computing security problem and its planning according to the cloud computing concepts and characters. He also stated that the data privacy and service availability in cloud computing are the key security problem. Single security method cannot solve the cloud computing security issues and many traditional and new technologies and strategies must be used together for protecting the total cloud computing system..[5]In NSA we can see the difference clearly between ECC and RSA and we are provided lot of valuable data to prove why ECC is over RSA. [6] In GeeksforGeeks they talked about How RSA algorithm works in Cryptography. The writer has given an example of asymmetric cryptography. Moreover, he talked about the idea of RSA is based on the fact that it is difficult to factorize a large integer. Encryption strength totally lies n the key size and if the key size be doubled or tripled in size, the strength of encryption increases exponentially. He also mentioned about the breaking down of keys which is not possible now but in near future it could be broken. [7]In DevCentral Wagon talked about both RSA and ECC and describes these algorithms mathematically. Here we can see RSA encryption, decription with example as well as ECC;s encryption, decryption with example. [8] Verma, Ojha discussed on some ECC algorithms and also have given mathematical explanations on the working of algorithms. [9] MS illustrated the introduction of ECC and how it is used in the implementation of digital signature and key agreement algorithms. He also implemented ECC and discussed on two finite fields, prime field and binary field. Also, he mentioned about the basics of prime and binary field arithmetic.10]Singh and other found out the beauty of using ECC instead of using RSA is that machines which make use of ECC are likely to consume less system resources thereby ameliorating the performance of the machines. In this paper, we experimentally evaluate the performance of RSA and ECC. [11] Malik divides his paper in five sections. Introduction to ECC, advantages of ECC's schemes and comparison with RSA, applications of ECC, embedded implementation of ECC, ECC implementation on fixed point Digital Signal Processor are described consecutively in section I, II, III, IV, V . [12]From off site backups protect your data, Eposlink we can get a visual sight of how cloud computing works .[14]Finally, Clercq discussed on Kerberos the basic protocol and it's five steps. [15]Some general information has been taken from Wikipedia [13]

Chapter 3

General Information on Cloud computing and Kerberos

3.1 Cloud Computing

Cloud computing is basically used as the term of computing or illustration which have emerged in the late 2000s, based on utility and consumption of computing resources. Cloud computing includes groups of remote servers and software networks that allow centralized data storage and online access to computer services or resources. Public, private or hybrid are three categories of cloud computing. In cloud computing, the word cloud (likewise stated as "the cloud") is utilized as a representation for "the Internet," so the expression cloud computing signifies "a sort of Server Internet-based figuring," where diverse administrations —, for example, servers, stockpiling and applications are conveyed to an association's PCs and gadgets through the Internet . Cloud computing is probably going to framework registering, a kind of figuring where unused handling cycles of all PCs in a system are tackles to take care of issues excessively serious for any independent machine. [4][5]

The objective of cloud computing is to apply conventional supercomputing, or superior processing power, regularly utilized by military and research offices, to perform many trillions of calculations for each second, in buyer situated applications, for example, money related portfolios, to convey customized data, to give information stockpiling or to influence substantial, immersive PC amusements. To do this, distributed computing utilizes systems of extensive gatherings of servers normally running minimal effort customer PC innovation with specific associations with spread information preparing tasks crosswise over them. This mutual IT framework contains enormous pools of frameworks that are connected together. Regularly, virtualization procedures are utilized to boost the energy of distributed computing.[4][5]

3.1.1. Types of Cloud Computing:

Cloud computing services will more often than not cover one of three things; they will either use the virtual servers to make a virtual IT, remotely facilitated programming, or a system stockpiling, with a chronicle of the information. Cloud computing is generally depicted in one of two ways. Either in view of the cloud area, or on the basis of service that the cloud is putting forth.

In view of a cloud area, we can characterize cloud as:[4]

- public,
- private,
- hybrid

The Public Cloud

Public cloud arrangements are promptly accessible from Google, Amazon, Microsoft, and others. Public cloud services give framework and services to people in general, and you, or your association, secure a bit of that foundation and system. Assets are shared by hundreds or thousands of individuals. Gmail and U of I Box are cases of public cloud services. Google revealed in April 2017 that it had 1 billion month to month clients. While your email account is secured by a secret key, the equipment on which it is put away is shared by more than 1 billion individuals.

The Private Cloud

Private cloud arrangements are devoted to one association or business, and regularly have significantly more particular security controls than does an open cloud. Numerous therapeutic workplaces, keeping money establishments and different associations who are required to meet government and state rules for information controls utilize a private cloud. Utilizing private distributed storage enables them to control exceptionally touchy information by meeting directions and industry-based criteria, regardless of whether that be restorative records, exchange insider facts, or other characterized data. Private cloud arrangements use foundation that is either possessed and controlled by the association, or they can authoritatively require those particular criteria be met by a merchant who deals with the framework.

The Hybrid Cloud

Hybrid cloud solutions are a blend of public and private clouds. This is a more complex cloud solution in that the organization must manage multiple platforms and determine where data is stored. An example of a hybrid cloud solution is an organization that wants to keep confidential information secured on their private cloud, but make more general, customer-facing content on a public cloud.

3.1.2. Services of Cloud Computing (SAAS, PAAS,IAAS) [4]

Cloud computing sorts on the premise of service sending models which let you pick the level of control over your data and sorts of services you have to give. There are three principle sorts of cloud computing services, some of the time called the cloud computing stack since they expand over each other.

So on the basis of services that the cloud is putting forth, we are talking about either:[4]

- SaaS (Software-as-a-Service)
- PaaS (Platform-as-a-Service)
- IaaS (Infrastructure-as-a-Service)

Infrastructure-as-a-service (Iaas):

The first cloud computing type is infrastructure-as-a-service (IaaS), which is used for Internet-based access to storage and computing power. The most basic category of cloud computing types, IaaS lets you rent IT infrastructure - servers and virtual machines, storage, networks and operating systems - from a cloud provider on a pay-as-you-go basis.

Platform-as-a-service (paas):

The second cloud computing type is platform-as-a-service (PaaS) which gives developers the tools to build and host web applications. PaaS is designed to give users access to the components they require to quickly develop and operate web or mobile applications over the Internet, without worrying about setting up or managing the underlying infrastructure of servers, storage, networks and databases.

Software-as-a-service (saas):

The third cloud computing type is software-as-a-service (SaaS) which is used for web-based applications. SaaS is a method for delivering software applications over the Internet where cloud providers host and manage the software applications making it easier to have the same application on all of your devices at once by accessing it in the cloud.

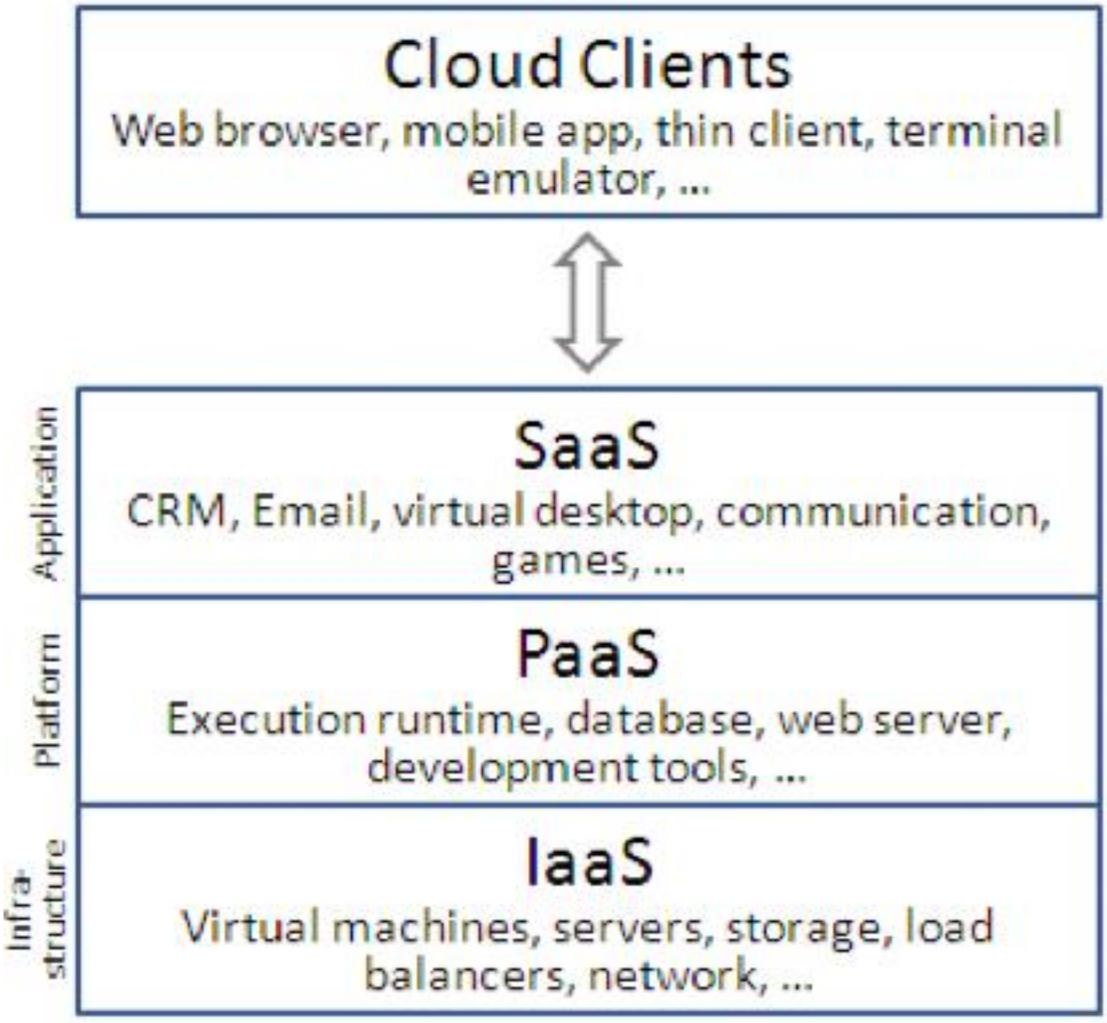

Cloud Clients
Web browser, mobile app, thin client, terminal emulator, ...
SaaS
CRM, Email, virtual desktop, communication, games, ...
PaaS
Execution runtime, database, web server, development tools, ...
IaaS
Virtual machines, servers, storage, load balancers, network, ...
Application
Platform
Infra-structure

Figure 3.1: Structural design of cloud [4]

3.1.3 Benefits of Cloud Computing

Cloud computing promises several attractive benefits for businesses and end users. Three of the main benefits of cloud computing includes:

- **Self-service provisioning:** End users can spin up computing resources for almost any type of workload on-demand.
- **Elasticity:** Companies can scale up as computing needs increase and then scale down again as demands decrease.

- **Pay per use:** Computing resources are measured at a granular level, allowing users to pay only for the resources and workloads they use. Cloud computing services are private, public or hybrid.

Private cloud administrations are conveyed from a business' server farm to interior clients. This model offers versatility and availability, while preserving administration, control and security. Inside clients either might possibly be charged for services through IT chargeback. In the general population cloud display, an outsider supplier conveys the cloud benefit over the Internet.Public cloud services are retailed on-request services to the clients, regularly incrementally or the hour. Clients just need to pay for the CPU cycles, stockpiling or data transfer capacity they utilize. Driving open cloud suppliers incorporate Amazon Web Services (AWS), Microsoft Azure, IBM/Soft Layer and Engine. Hybrid cloud is a blend of open cloud services and on-premises private cloud – with creation and computerization between the two. Organizations can be run mission-basic employments or touchy applications on the private cloud while utilizing general society cloud for curvaceous workloads that must scale on-request. The objective of hybrid cloud is to make a brought together, computerized, versatile condition which takes advantage of all that an open cloud framework can give, while as yet keeping up control over mission-basic information.

In spite of the fact that cloud computing has changed after some time, it has dependably been partitioned into three expansive service classifications: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). IaaS suppliers, for example, AWS supply a virtual server occasion and capacity, and also application program interfaces (APIs) that let clients relocate workloads to a virtual machine (VM). Clients have an assigned stockpiling limit and begin, stop, get to and design the VM and capacity as wanted. IaaS suppliers offer little, medium, substantial, additional huge, and memory-or register improved examples, notwithstanding modified occasions, for different workload needs. In the PaaS show, providers have advancement apparatuses on their frameworks. Clients get to those instruments over the Internet utilizing APIs, Web entries or entryway programming. PaaS is utilized for general programming improvement and numerous PaaS suppliers will have the product after it's created. Regular PaaS suppliers incorporate Salesforce.com's Force.com, Amazon Elastic Beanstalkand Google App Engine.SaaS is a sharing model that conveys programming applications over the Internet; these are frequently called Web services. Microsoft Office 365 is a SaaS offering for creation programming and email services. Clients can get to SaaS applications and services from any area utilizing a PC or cell phone that has Internet get to.

3.1.4. How Cloud Works

Consider an executive at a huge establishment. The particular responsibilities include making sure that all of the employees have the right access hardware and software they need to do their own jobs. Purchasing computers for everyone is not enough .it is essential buy software or software licenses to give employees the tools they require for their job. Whenever there is a new lease purchase more software or make sure that current software license permits another user. It's so worrying that find it difficult to go to sleep on a huge pile of money every night.

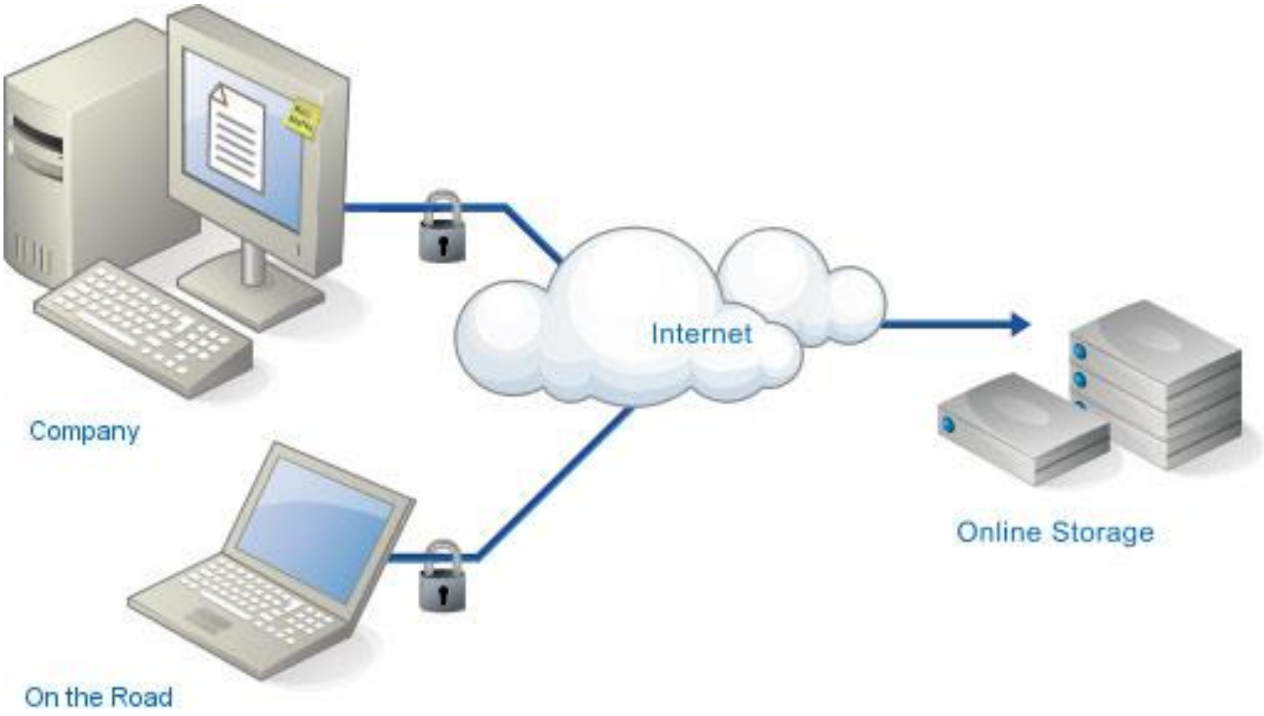


Figure 3.2: Cloud computing [14]

Soon, there may be a substitute for executives like client. Instead of installing a suite of software for each computer, client would have to load one application. That application will permit workers to log into a Web based service which hosts all of programs the user would need for his or her job. Remote machines owned by another company would run everything from e-mail to word processing to complex data analysis programs. It's called cloud computing, and it could change the whole computer industry.

Fig 1 shows that in a cloud computing system, there is an important work loading shift. Local computers do not have to do all the heavy lifting when it comes to runnable applications. The network of computers that made up the cloud handles them instead. Hardware and software demands on the users side is decrease. The only thing is that user's computer needs to be run in the cloud computing system's interface software, which can be as very easy as a Web browser, and the cloud's network takes care of the rest. There is a good chance users have already used some custom of cloud computing. If you have an e-mail account with a Web-based e-mail service like Hotmail, Yahoo! Mail or Gmail, then you have some experience with cloud computing. Instead of running an e-mail application on your computer, you log in to a Web e-mail account remotely. The software and storage for your account does not exist on your computer. it's on the service's computer cloud.

3.1.5. The disputes of cloud computing security are:

1. Authentication - assurance that communicating entity is the one claimed have both peer-entity & data origin authentication.
2. Access Control - prevention of the unauthorized use of a resource like computing.
3. Data Confidentiality –protecting of your data from unauthorized disclosure.
4. Data Integrity - guarantee that data received is as sent by an authorized entity.
5. Non-Repudiation - protection against denial by one of the parties in a communication

3.2 Kerberos

Kerberos is a PC organize verification convention that chips away at the premise of tickets to permit hubs conveying over a non-secure system to demonstrate their personality to each other in a safe way. The convention was named after the character Kerberos (or Cerberus) from Greek folklore, the fierce three-headed guard dog of Hades. Its designers pointed it fundamentally at a client– server model and it gives common verification—both the client and the server confirm each other's personality. Kerberos convention messages are ensured against listening in and replay attacks.

Kerberos expands on symmetric key cryptography and requires a trusted outsider, and alternatively may utilize public key cryptography amid specific periods of authentication. Kerberos utilizes UDP port 88 by default.

3.2.1. Terminologies of Kerberos Protocol

- AS= Authentication server to varify a client or user.
- TGS=Ticket Granting Server to generate a ticket to a client.
- AP=Application server.
- Principal client=Name given to a client in a realm.
- Principal Service = Name of the service needed by a client.
- Ip_List = IP address of the client.
- Database= Store the important data of its users to verify them.

3.2.2. Protocol Description:

The customer verifies itself to the Authentication Server (AS) which advances the username to a key distribution center (KDC). The KDC issues a ticket-granting ticket (TGT), which is time stamped and encodes it utilizing the ticket-granting ticket (TGS) secret key and returns the encrypted result to the client's workstation. This is done rarely, commonly at client logon; the TGT lapses eventually in spite of the fact that it might be straightforwardly recharged by the client's session administrator while they are signed in.

At the point when the customer needs to speak with another hub ("principal" in Kerberos speech) to some service on that node the customer sends the TGT to the TGS, which ordinarily shares an indistinguishable host from the KDC. Administration must be enlisted at TGT with a Service Principal Name (SPN). The customer utilizes the SPN to ask for access to this administration. In the wake of checking that the TGT is legitimate and that the client is allowed to get to the asked for benefit, the TGS issues ticket and session keys to the customer. The customer at that point sends the ticket to the service server (SS) alongside its administration request.

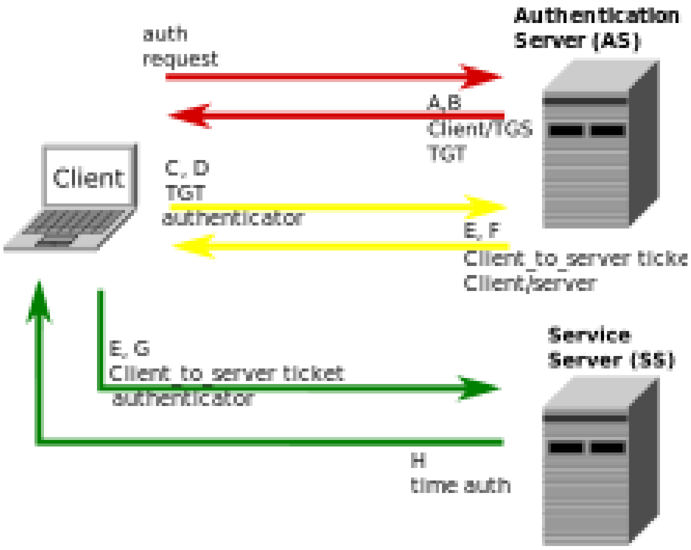


Figure 3.3: Kerberos negotiations [13]

3.2.3. Kerberos design assumption

There are some design assumptions we are going to discuss that are taken by the Kerberos designers at MIT.

- Kerberos dependably manages three elements: clients, servers and a set of security servers that intervene between the clients and the servers for verification.
- Time is trusted. This is because that Kerberos utilizes timestamps to secure against replay assaults.
- The client believes its workstation totally. This is because Kerberos reserves confirmation tokens on the customer side.

- The security server must be online constantly. Kerberos requires the accessibility of the security server with a specific end goal to create new Kerberos security tokens.
- The servers are stateless. Kerberos needs to restrain the measure of security principle - related data that is kept on the server side.
- Users' password time on user machine must be minimized. Kerberos looks at a user password as a weak secret -- it should be protected the best possible. One of the ways to do this is to limit its time on the user workstation. Another way is to create a key hierarchy.

.

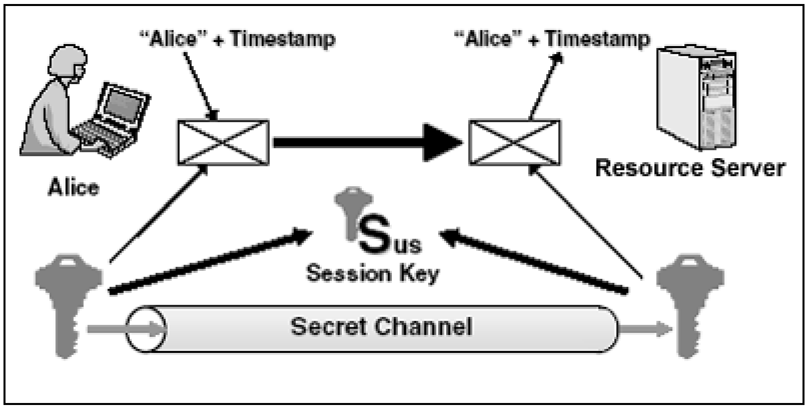


Figure 3.4: Kerberos authentication is based on symmetric key cryptography [15]

3.2.4. Steps in Kerberos Authentication Protocol

A short description on steps of Kerberos authentication protocol is given below.

Step01: C->AS(AS_REQ goes from client to Authenticator server)

To begin with customer logon to the system and demands for network service. The workstation sends a request message(AS_REQ) to the Validation Server for ticket granting ticket (TGT).The message design is as per the following:-

Message (1):- {client name, service name, Ip_list, timestamp} kuser

Step02: C->AS(AS_REQ goes from client to Authenticator server)

The Authentication Server checks the client's entrance rights in the client database and makes a Ticket Granting Ticket (TGT) and session key (SKTGS). This session key (SKTGS) is unique in relation to client logon session key (kuser).The message parcel is scrambled by the confirmation server utilizing customer logon session key and send it back to the customer workstation.

Message (2):- AS_REP= {{client name, Lifetime Principle service, SKTGS} Kuser {TGT} KTGS}

Here,

- KTGS :- Private key of the Ticket Granting Server.

- TGT:-{client Name, service name, Ip_list, Timestamp, lifetime, Stgs}

  - Time stamp :- Time stamp of KDC.
  - Life time :- max validity of the Ticket.

The workstation prompts incoming message for the client to enter a secret key or password and uses the password to decode the approaching messages parcel. After decryption, the client gets Ticket Granting Ticket (TGT).

Step 03: C->TGS(TGS_REQ goes from client to Ticket Authenticator server)

At the point when the client needs access to an administration, the workstation client application set up an authenticator sends a request to the Ticket Granting Service containing the customer name, Life time, and an authenticator encoded with the session key Stgs also, a TGT got in Step 2.

message (3):- TGS_REQ= {{client name, lifetime,
Authenticator} {TGT}} KTGS.

Here,

Authenticator= {Principal client, Timestamp}SKTGS

Step 04: TGS->C(reply given by Ticket Granting Server to client)

For the requested application server the Ticket Granting server (TGS) decodes the message packet, checks the demand, and makes an administration ticket (Tservice). The service ticket contains the customer name, ticket lifetime and alternatively the customer IP address. The TGS answered back with the service ticket to the client workstation. The replied message contains two copies of a server session key.

Message (4):- TGS_REP= {{Client Name, lifetime, Timestamp, SKservice} SKTGS {Tservice} Kservice.

Here,

- Kservice= server secret key.
- SKservice is the service session key shared between client and server.
- Tservice is the service ticket having the following contents:-{Client name, lifetime, Principle service, Ip_list, Timestamp, SKservice}

Step 05: C->AP(AP_REQ goes from client to server)

The customer now sends a requesst (AP_REQ) to the application server containing the service ticket and an authenticator. The administration confirms the request by decoding the session key. Then the server verifies weathe the ticket and authenticator is matching or not. If it is matched then it grants access to the service.

Message(5):- AP_REQ= {Authenticator, Tservice} Kservice.

Here,

Authenticator={Client Name, Timestamp}SKservice]

Step 06:- AP->C(Reply given by application server)

If mutual authentication is required, then the server will reply with a server authentication message. It is an optional step. If all the above steps are successfully executed then a client will use the requested services.[Emam,2009][Anita narwal,2015]

3.2.5. Bringing it all together

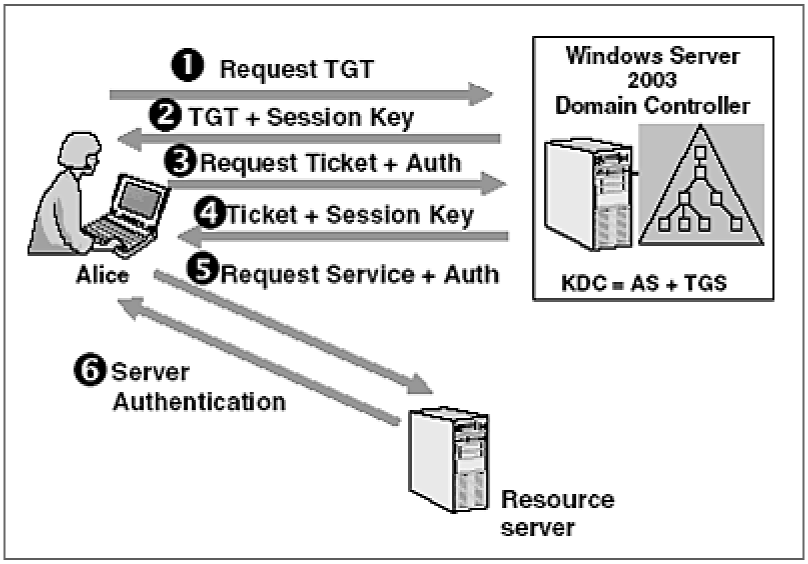


Figure 3.5: The full authentication process [15]

Chapter 4

Algorithms

4.1 Threshold Cryptology:

The idea of threshold cryptography is to shield information (or computation) by way of fault-tolerantly distributing it amongst a cluster of cooperating computer systems. First bear in mind the fundamental trouble of threshold cryptography, a problem of secure sharing of a secret. A secret sharing scheme permits one to distribute a piece of mystery information among numerous servers in a manner that meets the subsequent necessities: (1) no organization of corrupt servers (smaller than a given threshold) can determine out what the secret's, although they cooperate; whilst it will become important that the secret statistics be reconstructed, a big enough number of servers (a number large than the above threshold) can constantly do it.

A very useful extension of secret sharing is function sharing. Its primary idea is that a tremendously sensitive operation, together with decryption or signing, may be achieved by means of a group of cooperating servers in such a way that no minority of servers is able to perform this operation through themselves, nor would they be able to prevent the other servers from acting the operation while it's miles required.(1)

Purpose for the usage of threshold cryptography is to offer greater protection to the key used by secret share scheme. On this scheme data D is divided into n portions and expertise of a few pieces m is permits to derive secret data D. Knowledge of any portions m-1 makes secret data D absolutely undetermined. This sort of scheme is referred to as a (m, n) threshold scheme. This scheme is without problems computable whilst it has important data available. This is a safe and comfort technique to provide protection to key.(1)

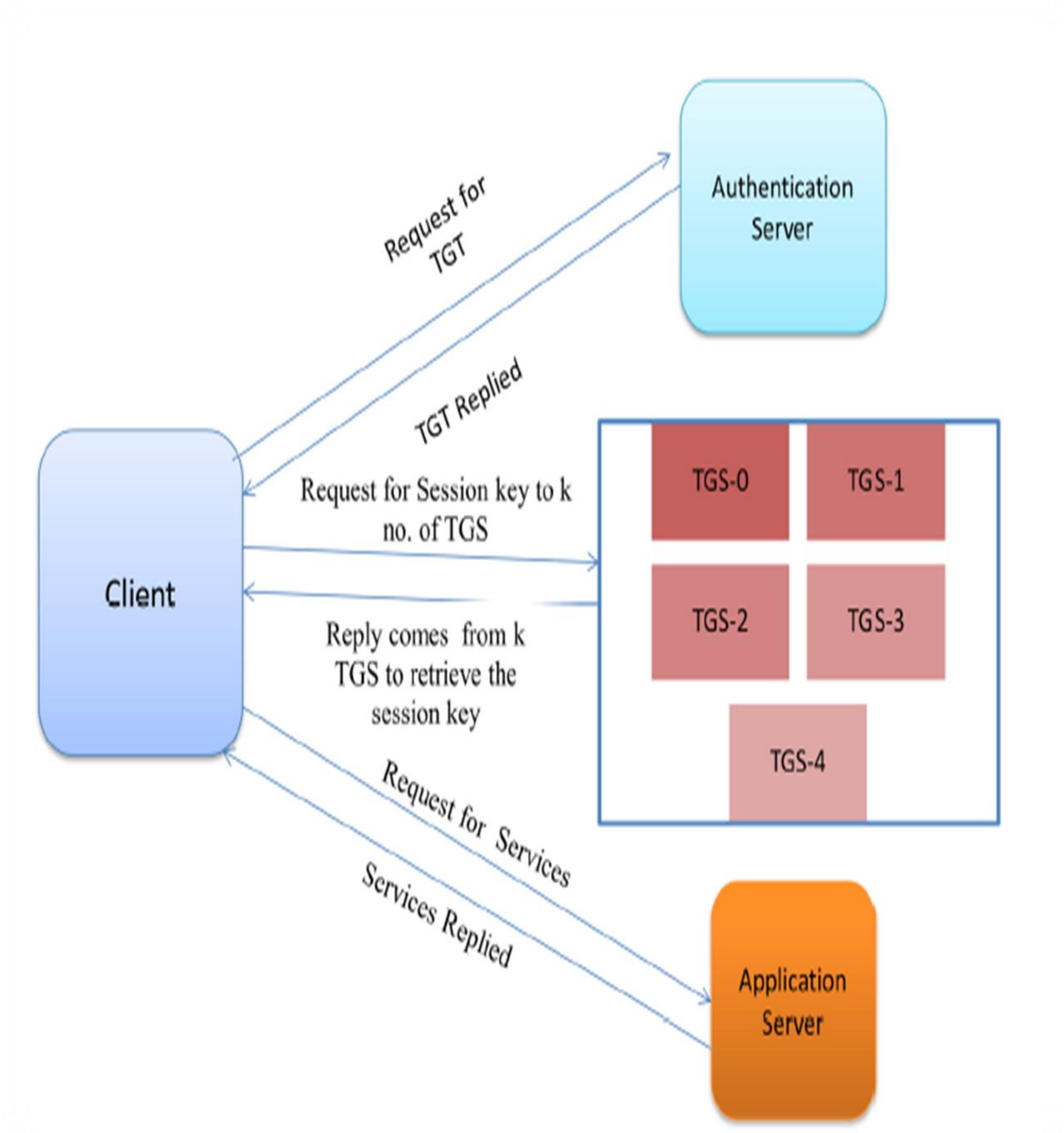


Figure 4.1: Threshold Cryptography Mechanism [1]

4.1.1 Mathematical explanation of Threshold Cryptography Algorithm:

Let us take (m, n) threshold scheme to distribute our secret S. Among them, choose random m-1 coefficients and let they are as $p_0, p_1, p_2 \ldots\ldots\ldots p_{m-1}$. Assume that p_0=S. We divide our secret S by picking a random degree polynomial as follows, (1)

Q(x) = $p_0 + p_1x + p_2x^2 + \ldots\ldots p_{m-1}x^{m-1}$

The subsets of m number of pairs can find secret s , by using interpolation

Method. The secret is the constant term of that interpolation equation which is p_0. To evaluate the above expression,

Let's S=2469, n=7, m=3

Randomly selected two numbers : p_1=207 , $p_2 = 86$ which are used to make the polynomial:

Q(x) = 2469 + $207x + 86x^2$……………………….(1)

Now, from equation (1) we can get 7 separate points for x=1, 2, 3 ………7. When x=1 then Q(1)=2762

Similarly, When x=2 then Q(2)=3227

When x=3 then Q(3)=3864

When x=4 then Q(4)=4673

When x=5 then Q(5)=5654

When x=6 then Q(6)=6807

When x=7 then Q(7)=8132

So seven points are obtained from the polynomial: (1, 2762); (2, 3227); (3, 3864); (4, 4673); (5, 5654); (6, 6807); (7, 8132). So that each participant can get different single point (both x and Q(x)).

4.1.2 To reform the secret:

In order to reform the secret S, m points will be enough. Let m=3 and consider (3, 3864); (5, 5654); (7, 8132). It is possible to form Q (x) again by using Lagrange's polynomial, and the value of S can also be retrieved which is same as before. Let us consider (x0, y0) = (3, 3864); (x1, y1) = (5, 5654); (x2, y2) = (7, 8132). Lagrange's polynomials can be computed as :

$$L_0 = \frac{x-x_1}{x_0-x_1} . \frac{x-x_2}{x_0-x_2} = \frac{x-5}{3-5} . \frac{x-7}{3-7} = \frac{1}{8} . x^2 - 1\tfrac{1}{2} . x + 4\tfrac{3}{8}$$

$$L_1 = \frac{x-x_0}{x_1-x_0} . \frac{x-x_2}{x_1-x_2} = \frac{x-3}{5-3} . \frac{x-7}{5-7} = -\frac{1}{4} . x^2 - 2\tfrac{1}{2} . x + 5\tfrac{1}{4}$$

$$L_2 = \frac{x-x_0}{x_2-x_0} . \frac{x-x_1}{x_2-x_1} = \frac{x-3}{7-3} . \frac{x-5}{7-5} = \frac{1}{8} . x^2 - x + 1\tfrac{7}{8}$$

F(x) = $\sum_0^2 (x)\, y_i l_i$ ....(2)

Now by using the above equation (2), we can reform our secret in this way,

F(x) = 3864($\frac{1}{8} . x^2 - 1\tfrac{1}{2} . x + 4\tfrac{3}{8}$) + 5654($-\frac{1}{4} . x^2 - 2\tfrac{1}{2} . x + 5\tfrac{1}{4}$) +8132($\frac{1}{8} . x^2 - x + 1\tfrac{7}{8}$)

= Q(x) = 2469 + $207x + 86x^2$

After necessary calculations we get the same equation as equation no (1) Remember that the secret is the free coefficient, which means that S= 2469, and as it is similar to the equation (1).

RSA and ECC are known as the most effective PKC among all uneven encryption calculations. They gloat an extensive number of benefits in examination with other cryptosystems. [Kumar,2006]

4.2 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.

The idea of RSA is based on the fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is multiplication of two large prime numbers. And private key is also derived from the same two prime numbers. So if somebody can factorize the large number, the private key is compromised. Therefore encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024 bit keys could be broken in the near future. But till now it seems to be an infeasible task.[2]

4.2.1. Mechanism behind RSA algorithm:

The components of the RSA cryptosystem are:

p = random prime number

q = random prime number

n = p * q

Φ(n) = (p-1) * (q-1)

e = number between 1 and Φ(n)

d = e^{-1} (modulo Φ(n))

Public Key = key pair (e, n)

Private Key = key pair (d, n)

4.2.2. RSA Encryption:

RSA use public key to encrypt data or messages into cipher text by using the following formula,

$$\mathbf{m^e \bmod n}$$

4.2.3. RSA Decryption:

RSA use private key to decrypt cipher text into original text by using the following formula,

$$\mathbf{c^d \bmod n}$$

4.2.4. RSA Working Example:

Now that all the pieces are in place and we have the formulas needed to encrypt and decrypt, let's run through a working example of the RSA public key cryptosystem. We will start with the random prime numbers of 11 and 13. Using all the calculations above, we have:

$p = 11$

$q = 13$

$n = 11 \text{ x } 13 = 143$

$\Phi(n) = 10 \text{ x } 12 = 120$

$e = 7$ (it turns out that 7 is between 1 and 120, and the GCD of 7 and 120 is 1…so, it fits all the criteria to be our public key value)

$d = 7^{-1} \text{ (mod } 120) = 103$

The public key is represented by the key pair (7, 143)

The private key is represented by the key pair (103, 143)

Let's start with a plain text value of "9" and let's encrypt it because, you know, it's super-sensitive information. Using the key values that were generated above, we find that the encrypted value is:

$9^7 \text{ mod } 143 = 48$

So, the encrypted value for "9" is "48" using all the RSA numbers we chose above. Obviously, the encrypted values will change given different values for p, q, n, etc.

To decrypt the value, we use our handy-dandy decryption formula and find that:

$48^{103} \text{ mod } 143 = 9$

And, just like that, we are back at our original value of 9.

4.3 Elliptic Curve Cryptography (ECC):

Elliptic Curve Cryptography (ECC) is a public key cryptography. In public key cryptography each user or the device taking part in the communication generally have a pair of keys, a public key and a private key, and a set of operations associated with the keys to do the cryptographic operations. Only the particular user knows the private key whereas the public key is distributed to all users taking part in the communication. Some public key algorithm may require a set of predefined constants to be known by all the devices taking part in the communication.[1]

The elliptic curves used in cryptography today are typically defined by the following algebraic function:

$$\mathbf{y^2 = x^3 + ax + b}$$

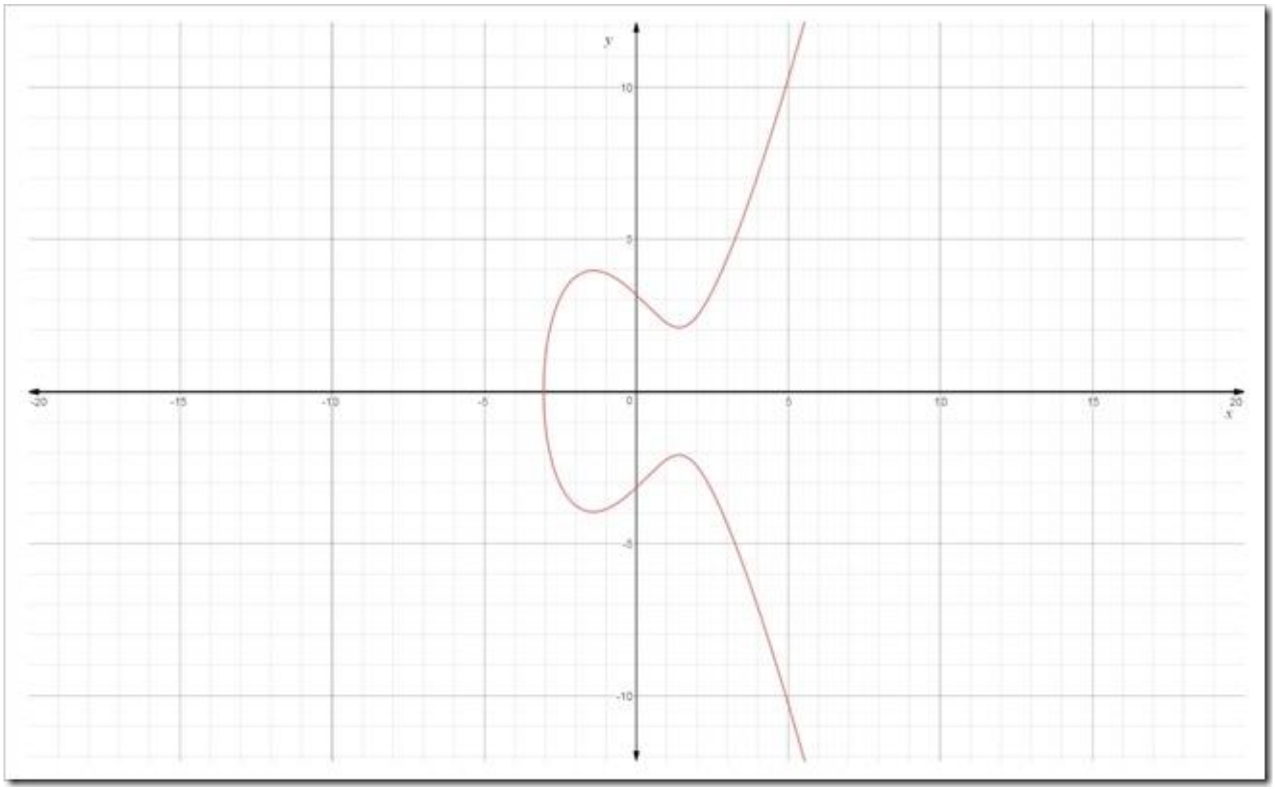


Figure 4.2: An elliptic curve [8]

To understand this "hopping around the curve", let's begin with a few interesting characteristics about elliptic curves as well as a concept known as Point Addition. Elliptic curves have symmetry about the x-axis, and any non-vertical line will intersect the curve in at most 3 points.

4.3.1. Point Addition:

Point Addition is an operation on an elliptic curve that allows you to start with one point and ultimately arrive at another point on the curve. Here's how Point Addition works: given two points on the curve (P and Q), draw a straight line through them and intersect the curve at a third point (called -R). Then, follow the value for -R along a vertical line until you intersect the curve again. This intersecting point is the value for R. So, P+Q = R

4.3.2. Point Addition technique [5]:

Consider two distinct points J and K such that J = (xJ, yJ) and K = (xK, yK) Let L = J + K where L = (xL, yL), then xL = s2 - xJ – xK mod p yL = -yJ + s (xJ – xL) mod p s = (yJ – yK)/(xJ – xK) mod p, s is the slope of the line through J and K. If K = -J i.e. K = (xJ, -yJ mod p) then J + K = O. where O is the point at infinity. If K = J then J + K = 2J then point doubling equations are used. Also J + K = K + J

The graph below shows an example of Point Addition

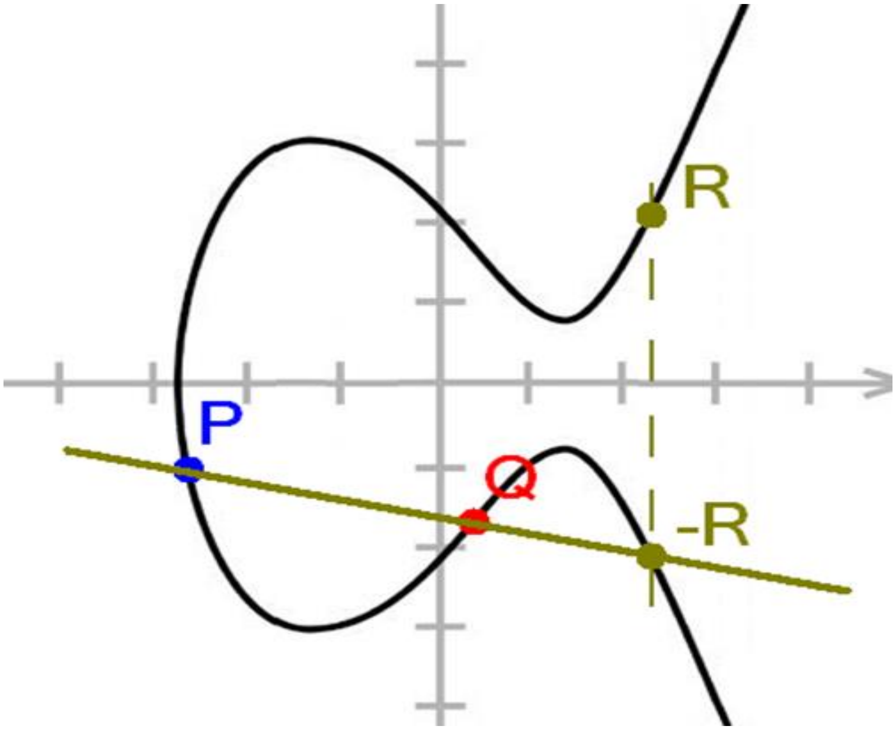


Figure 4.3: Point addition [8]

4.3.3. Point Doubling

Point Doubling is similar to Point Addition except that in point doubling, you add P to itself rather than add P to another point on the curve. In this case, we have to draw the tangent line to the point (P) and then let the tangent line intersect the curve at another point. At the intersecting point, we follow along a vertical line until we intersect the curve again (exactly the same concept as the P + Q operation above). At that intersecting point, we will find the value for P + P. The Point Doubling operation is shown on the graph below.[9]

4.3.4. Point Doubling technique: Consider a point J such that J = (xJ, yJ), where yJ ≠ 0 Let L = 2J where L = (xL, yL), Then xL = s2 – 2xJ mod p yL = -yJ + s(xJ - xL) mod p Elliptic Curve Cryptography – An Implementation Tutorial 5 s = (3xJ 2 + a) / (2yJ) mod p, s is the tangent at point J and a is one of the parameters chosen with the elliptic curve If yJ = 0 then 2J = O, where O is the point at infinity.[9]

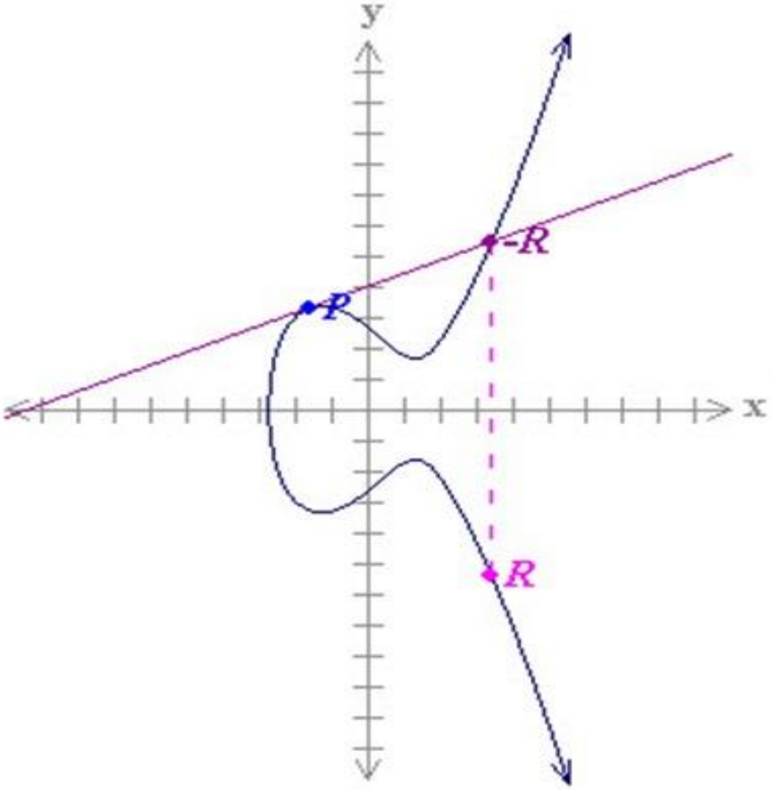


Figure 4.4: Point doubling [8]

4.3.5. Point multiplication

Point multiplication is the central operation in ECC. In point multiplication a point P on the elliptic curve is multiplied with a scalar k using elliptic curve equation to obtain another point Q on the same elliptic curve. i.e. k*P=Q Point multiplication is achieved by the above two operation which are point addition and point doubling . Here is a simple example of point multiplication. Let P be a point on an elliptic curve. Let k be a scalar that is multiplied with the point P to obtain another point Q on the curve. i.e. to find Q = k*P. If k = 23 then k*P = 23*P = 2(2(2(2P) + P) + P) + P

let’s go over the values we need in order to fully define the ECC cryptosystem. [8] These are:

Curve equation: $y^2 = x^3 +ax + b$

p: Specifies the finite field that the curve will be defined over (modulo value)

a: Coefficient that defines the curve

b: Coefficient that defines the curve

G: Generator point on the curve. This is the point where all the Point operations begin.

n: Order of G. The number of Point operations on the curve until the resultant line is vertical.

h: Cofactor – the number of points on the elliptic curve divided by the order of G (ideally this value is 1 or very close to 1)[8]

4.3.6. Cryptographic system:

An overview of EC cryptographic algorithms for key agreement and digital signature are explained below.

ECDH – Elliptic curve Diffie **Hellman:**

ECDH, a variant of DH, is a key agreement algorithm. For generating a shared secret between A and B using ECDH, both have to agree up on Elliptic Curve domain parameters. An overview of ECC cryptographic algorithms for key agreement and digital signature are explained here.[9]

4.3.7. Key Agreement Algorithm*:*

For establishing shared secret between two device A and B

Step1. Let dA and dB be the private key of device A and B respectively, Private keys are random number less than n, where n is a domain parameter.

Step2. Let QA = dA*G and QB = dB*G be the public key of device A and B respectively, G is a domain parameter

Step3. A and B exchanged their public keys

Step4. The end A computes K = (xK, yK) = dA*QB

Step5. The end B computes L = (xL, yL) = dB*QA

Step6. Since K=L, shared secret is chosen as xK [9]

4.3.8. ECDH - Mathematical Explanation

To prove the agreed shared secret K and L at both devices A and B are the same From Step2, Step4 and Step5

K = dA*QB = dA*(dB*G) = (dB*dA)*G = dB*(dA*G) = dB*QA = L

Hence K = L, therefore xK = xL Since it is practically impossible to find the private key dA or dB from the public key QA or QB, its not possible to obtain the shared secret for a third party. [9]

4.3.9. A Working Example of ECC

Here are the values for our ECC cryptosystem using the Diffie Hellman key agreement protocol [8]:

Curve: $y^2 = x^3 + 2x + 2 \pmod{17}$

p: 17 (notice this is a prime number, so this is considered a “prime” curve)

a: 2

b: 2

G: (5,1)

n: 19

To find n, you Point Double/Point Add starting from G until you reach a point at infinity. That is, the operations continue until the resultant line is vertical. In this case, n = 19. Here are the first few operations for this particular curve; starting at the Generator Point (5,1):

2G = G + G = (6,3) (This point and the remaining points can be found using Point Doubling)

3G = 2G + G = (10,6)

4G = 3G + G = (3,1)

…

19G = ∞

h = 1 (which is the ideal value for h)

Now we are ready to start computing values for α and β.

Mukti picks a value for α. The value for α must be between 1 and n-1 (18).

α = 3

Next, Mukti computes the value for A:

A = α * G = 3G

A = (10,6)

Notice that the value for A is not a single number. Rather, it is the point on the curve represented by Point Doubling/Point Addition operations conducted α times.

Urm picks a value for β. The value for β must be between 1 and n-1 (18).

β = 9

B = β * G = 9G

B = (7,6)

They share the values A and B with each other and are then ready to both compute the value for P.

Urmi computes P = β * A = β * 3G = 9 * 3G = 27G. Because the order of the curve (n) is 19, 27G reduces to 8G. If the value for P results in a number higher than the order of the curve, you use the modulo operation to find the resulting value. In this example, 27 mod 19 = 8. So, 27G becomes 8G because 27 is larger than 19.

Mukti computes P = α * B = α * 9G = 3 * 9G = 27G. Mukti uses the same logic as Urmi in reducing 27G to 8G.

So, P = 8G = (13,7)

Now, Mukti and Urmi both have the point (13,7) as their shared secret, and the man in the middle has no idea what the value for P is.

Chapter 5

Our proposed model

5.1 Our Model:

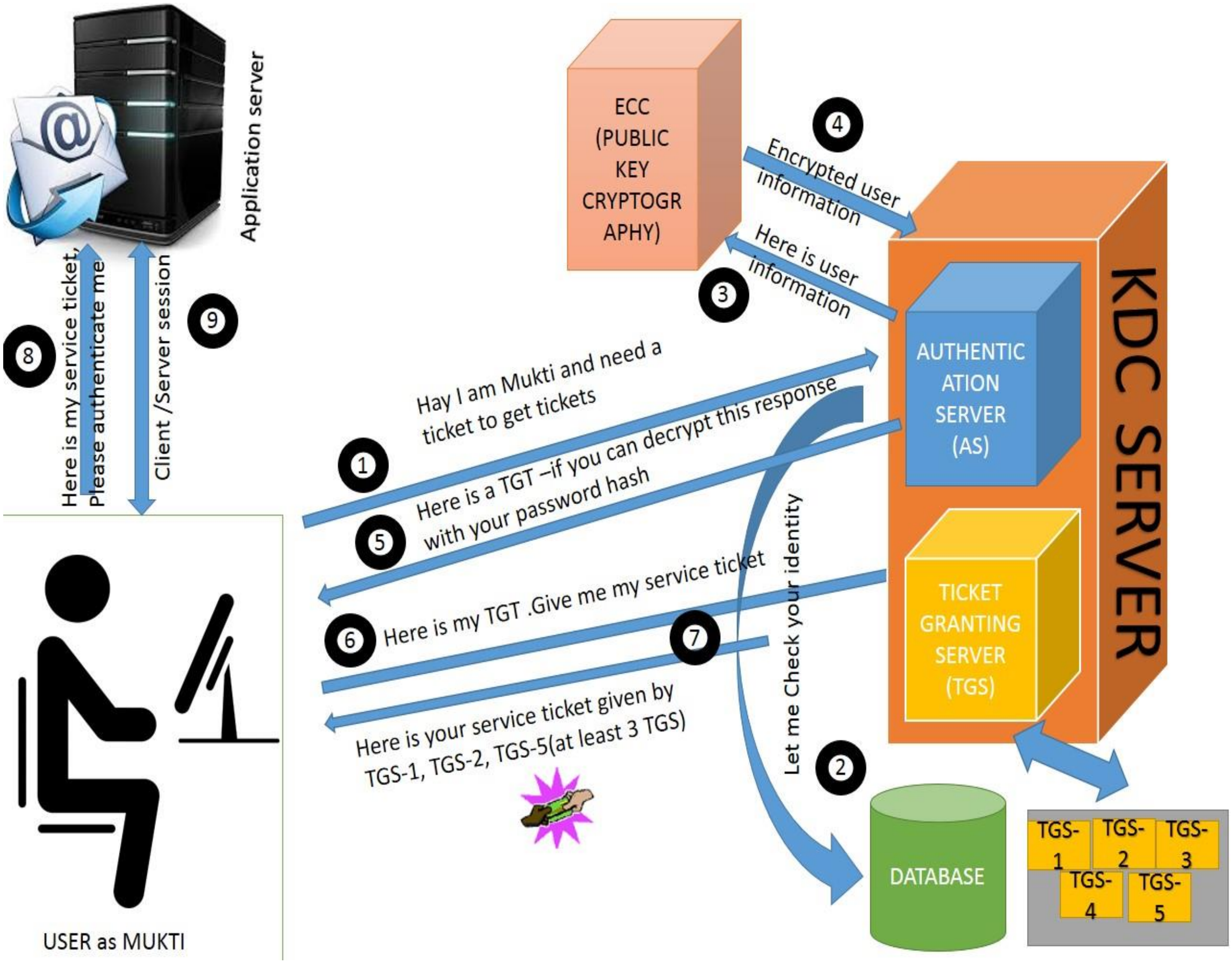

Application server
ECC (PUBLIC KEY CRYPTOGRAPHY)
4
Encrypted user information
Here is user information
3
KDC SERVER
AUTHENTICATION SERVER (AS)
TICKET GRANTING SERVER (TGS)
8
Here is my service ticket, Please authenticate me
Client /Server session
9
Hay I am Mukti and need a ticket to get tickets
1
Here is a TGT –if you can decrypt this response with your password hash
5
Here is my TGT .Give me my service ticket
6
7
Let me Check your identity
Here is your service ticket given by TGS-1, TGS-2, TGS-5(at least 3 TGS)
2
DATABASE
TGS-1
TGS-2
TGS-3
TGS-4
TGS-5
USER as MUKTI

Figure 5.1 : Our proposed model

5.2 Model description

In this model, when a user like Mukti wants to access any application server, she has to go through some steps through which she can make her access to the expected server more securely.

Step 1: First Mukti has to make a request to KDC (Kerberos Distribution Center) by entering her user name, password or her identity so that she can access the TGS (Ticket Granting Server) for further access.

Step 2: After getting the user request, KDC will check it’s database whether the recent requested entry is exist or not. If exist then KDC will reply with a simple acknowledgement with a session key and permission key(ticket) which are in encrypt form that you are now allow to collect tickets from TGS server else do not allow to access further.

Step-3: If Mukti is valid user then she would be able to decrypt that permission key by using ECC public key cryptography, otherwise not.

Step-4: When Mukti get the permission to make a request to TGS server, she immediately tries to connect with TGS server with her permission key for collecting pass tickets.

Step-5: After getting request from Mukti, TGS server (here TGS server is chunked into n no of servers and at least k no of TGS server is responsible for one access request) will send acknowledgement one by one. Simultaneously The KDC will send a message to that particular application server that someone is going to access you.

Step-6: If Mukti get reply from at least k no of TGS server, then she is now a valid user to get access to an application server because already application server has been informed about Mukti as valid user.

5.3 Our Work Flowchart:

AS= Authentication Server

TGT=

Ticket Granting Ticket

TGS= Ticket

Granting Server

n= no of parts of keys

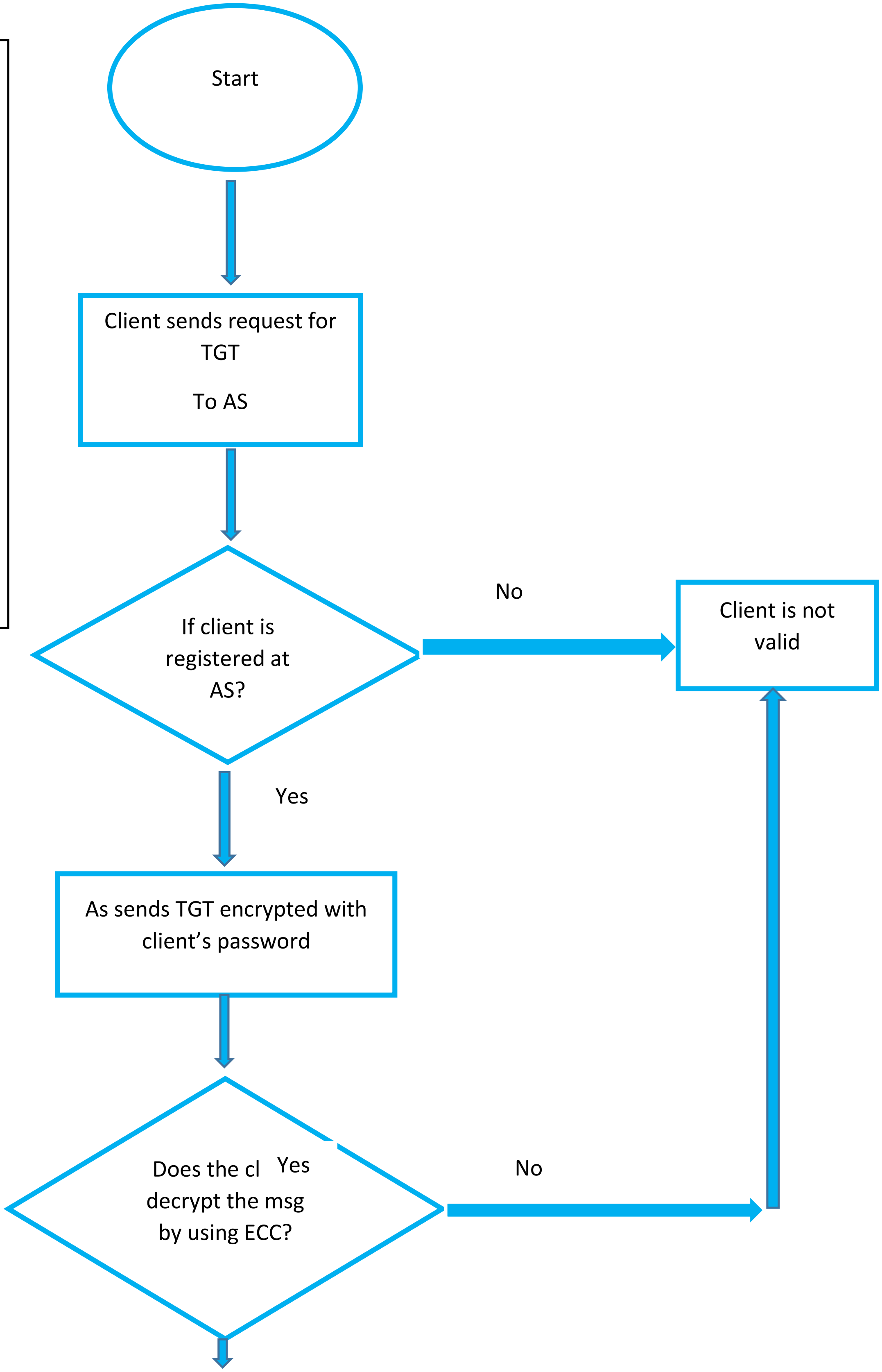
k= minimum no. of keys to decrypt data

t= waiting time

r= no of reply

req= Request

msg= Message


Start
Client sends request for TGT
To AS
If client is registered at AS?
No
Client is not valid
Yes
As sends TGT encrypted with client's password
Does the cl Yes
decrypt the msg
by using ECC?
No

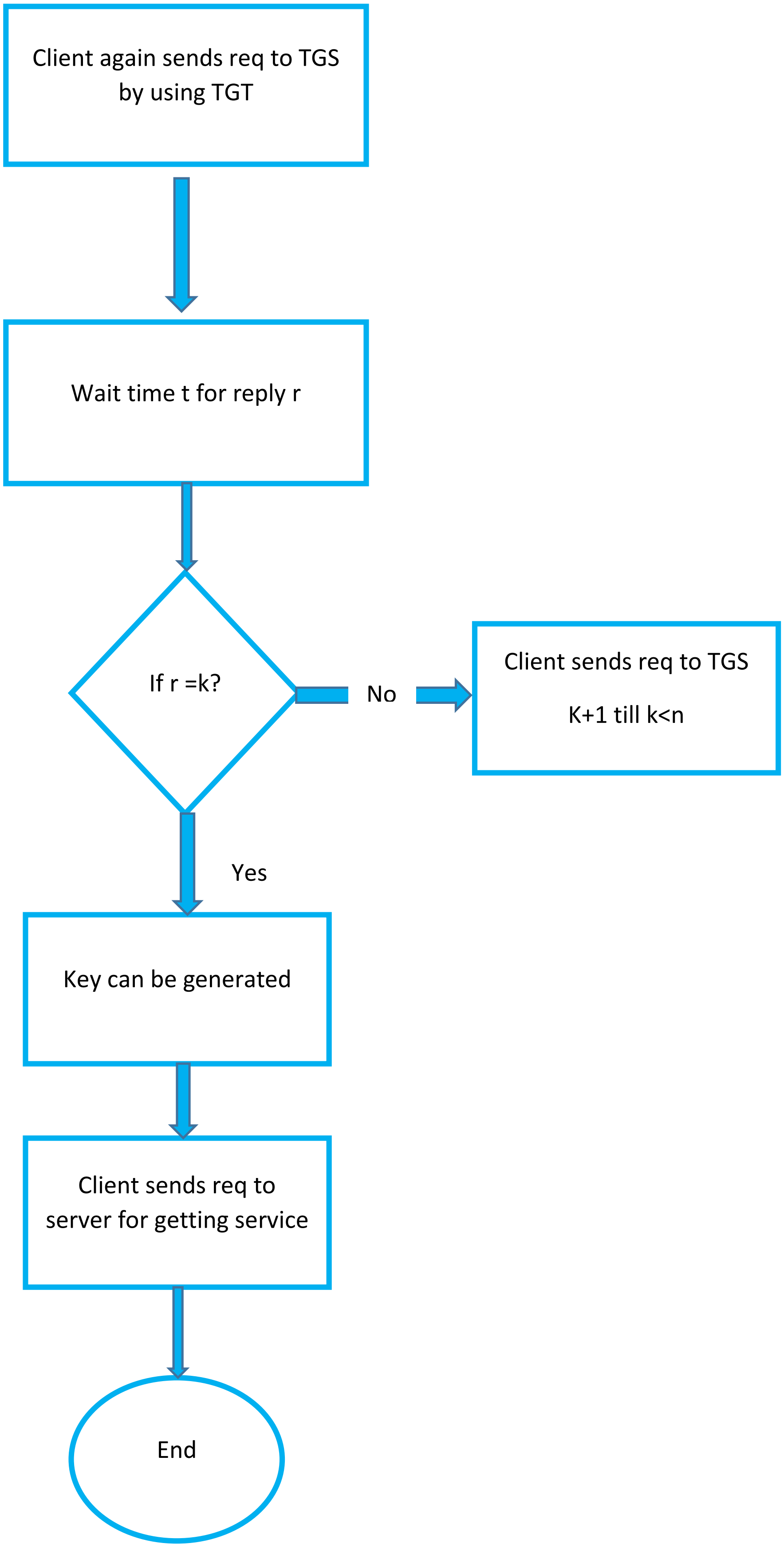


Figure 5.2: Flow chart of new version of Kerberos

Chapter 6

Implementation of Kerberos authentication protocol:

To implement Kerberos authentication protocol we have used here UBUNTU 16.04.3 LTS. After installing UBUNTU 16.04 we have configured the hostname of this server pc as ubuntu.domain.local

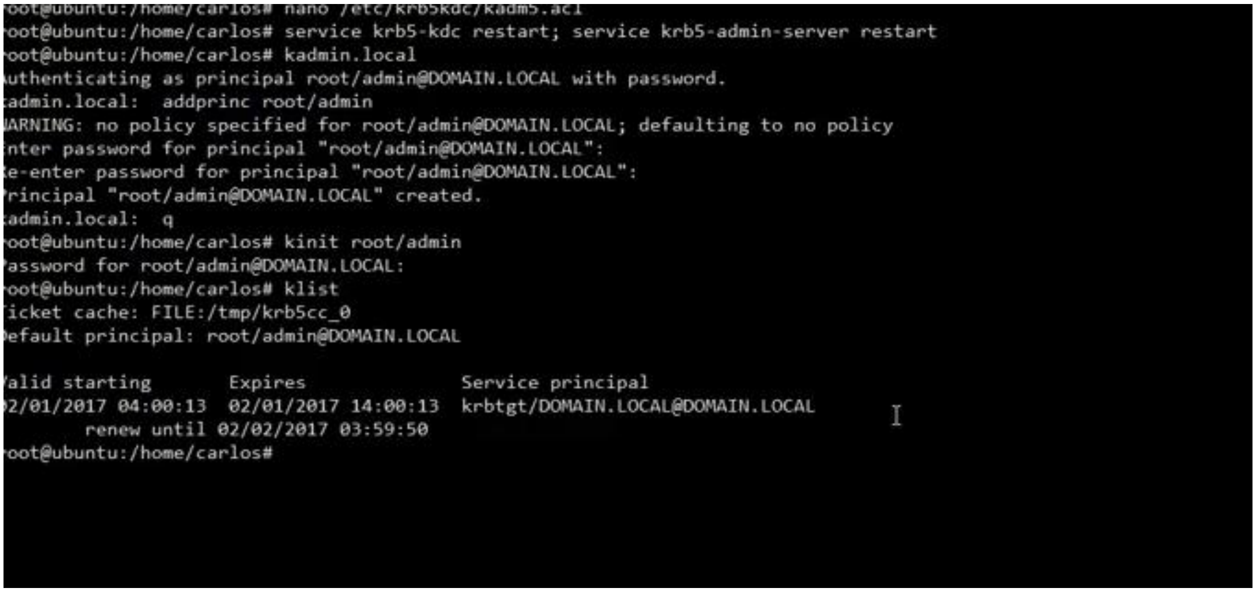
To implement the basic Kerberos we have installed Krb5-Kdc in our server. In this process we have used a default realm and Kerberos KDC servers for this default realm . After installing kdc we have configured the config file. We have also installed krb5-admin-server. In this process admin-server initialized database for realm DOMAIN.LOCAL

There we got prompted for KDC database master key. Now we can use this authentication protocol simply using kinit command. We can look at the tgt ticket just applying klist command after using kinit command.

Klist shows us krbtgt which contains valid starting time, expire time and service principal.

To implement Kerberos in apache server at first we have to install apache2 plugin. After that we have to configure the auth-kerberos.conf file. This will also let us use Kerberos for apache.

Now we can use Kerberos also for apache using Mozilla Firefox from client PC.



```
oot@ubuntu:/home/carlos# nano /etc/krb5kdc/kadm5.acl
oot@ubuntu:/home/carlos# service krb5-kdc restart; service krb5-admin-server restart
oot@ubuntu:/home/carlos# kadmin.local
uthenticating as principal root/admin@DOMAIN.LOCAL with password.
admin.local:  addprinc root/admin
ARNING: no policy specified for root/admin@DOMAIN.LOCAL; defaulting to no policy
nter password for principal "root/admin@DOMAIN.LOCAL":
e-enter password for principal "root/admin@DOMAIN.LOCAL":
rincipal "root/admin@DOMAIN.LOCAL" created.
admin.local:  q
oot@ubuntu:/home/carlos# kinit root/admin
assword for root/admin@DOMAIN.LOCAL:
oot@ubuntu:/home/carlos# klist
icket cache: FILE:/tmp/krb5cc_0
efault principal: root/admin@DOMAIN.LOCAL

alid starting       Expires              Service principal
2/01/2017 04:00:13  02/01/2017 14:00:13  krbtgt/DOMAIN.LOCAL@DOMAIN.LOCAL
        renew until 02/02/2017 03:59:50
oot@ubuntu:/home/carlos#
```

Fig 6.1. Implementation

So let's say our ftp server's name is ftp.domain.local which is basically under the default realm DOMAIN.LOCAL. when the user will send request to access the ftp server the user will create a packet which will include the username and additional credentials like password , ip- address , timestamp etc. These credentials also be encrypted with secret key and the client also send this secret key along this packet to KDC . KDC will decrypt this packet with the secret key which was sent by the client. Then KDC will check his database if there is any user name for this realm in his database. If username found then KDC will encrypt username, ip-address, password, timestamp etc. KDC never shares his secret key. KDC will send this packet to client. Then client again will make a packet which will contain username, password, ip-address, time stamp and again this client will encrypt all these credentials and send the KDC's packet and the clients packet both with the client secret key or session key. Now KDC will decrypt the clients packet send by client with session key and KDC's packet with KDC's secret key .KDC will match the information of both packets. IF all the information are okay and satisfactory then KDC will generate the tgt and will send it to the client to access FTP server.

Chapter 7

Data analysis and Result:

In this section ,we have shown some experimental data through which we can understand the comparison between ECC and RSA cryptography, and also in other part efficiency of threshold cryptography.

7.1 Why ECC over RSA:

In the existing Kerberos , there are RSA algorithm which is a public key cryptography but in our new version of Kerberos we are going to use another public key cryptography. There are some valid reasons behind using ECC as a public key cryptography. They are as follows,[12]

7.1.1. More Complex than RSA:

In spite of multiplication or exponentiation in finite field, ECC uses scalar multiplication. Solving Q=k.P (utilized by ECC) is more difficult than solving factorization (used by RSA) and discrete logarithm (used by Diffie-Hellman (DH)), much stronger than other public key agreement and signature authentication methods.

7.1.2 Involvement of Less Number of Bits [6]

ECC requires much lesser numbers and thus less number of bits for its operation.

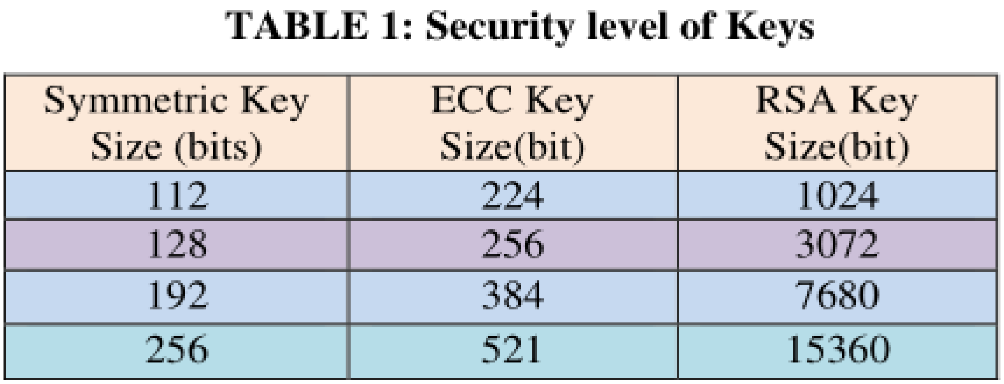


TABLE 1: Security level of Keys

Symmetric Key Size (bits)	ECC Key Size(bit)	RSA Key Size(bit)
112	224	1024
128	256	3072
192	384	7680
256	521	15360

Figure7.1:Table of Security Level of Keys [11]

To clarify the comparison, illustrates the data in TABLE I with bit ratio.

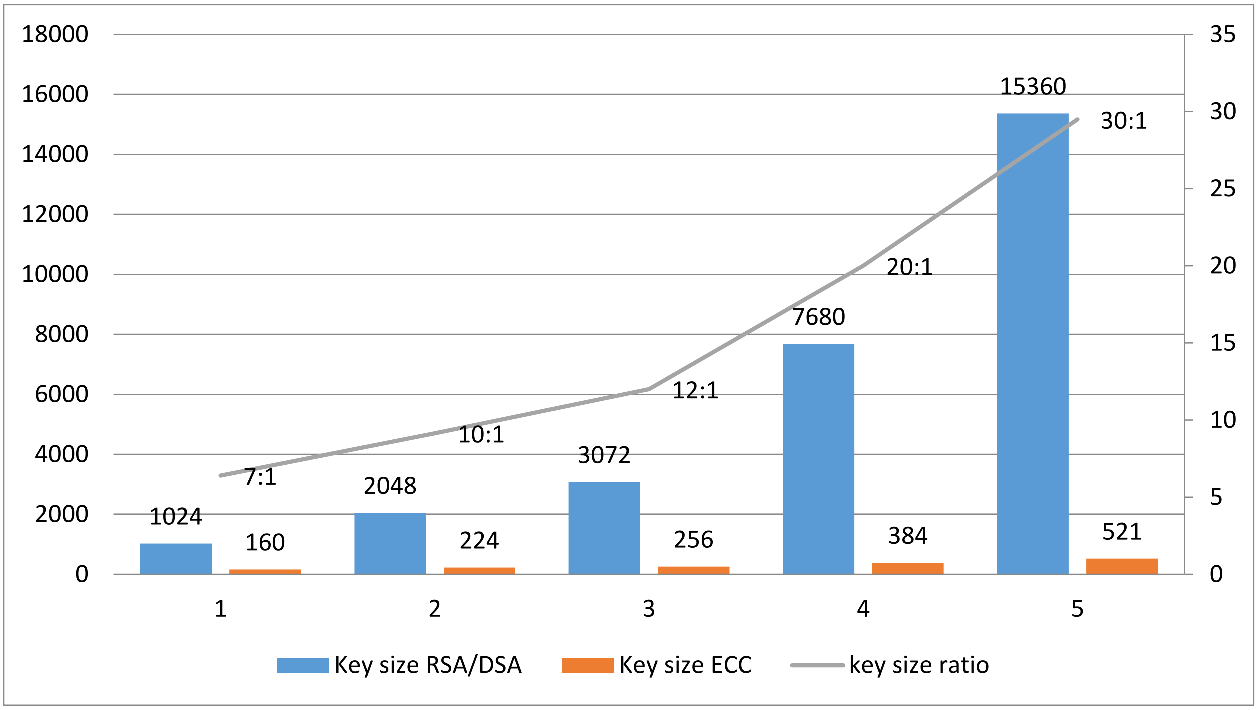


Figure 7.2: Graph of the Table 1

7.1.3. Computational Efficiency:

Implementing scalar multiplication in software and hardware is much more feasible than performing multiplications or exponentiations in them. As ECC makes use of scalar multiplications so it is much more computationally efficient than RSA and Diffie-Hellman (DH) public schemes. So we can say without any doubt that ECC is the stronger and the faster (efficient) amongst the present techniques. [12]

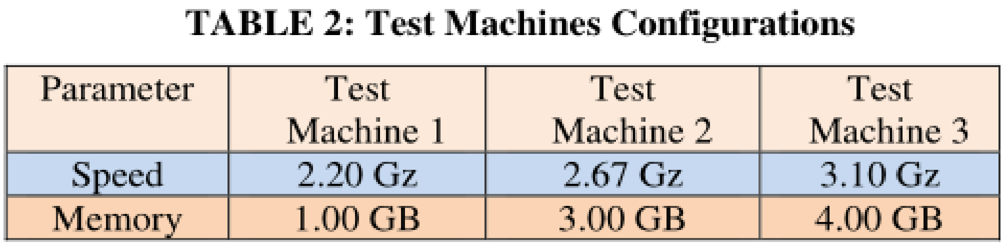


TABLE 2: Test Machines Configurations

Parameter	Test Machine 1	Test Machine 2	Test Machine 3
Speed	2.20 Gz	2.67 Gz	3.10 Gz
Memory	1.00 GB	3.00 GB	4.00 GB

Figure 7.3: Test Machines configurations table [12]

7.1.4. Public Key Generation Time

ECC requires less time to generate public key compared to RSA. In the following table we have shown that for a particular message there are huge time difference between ECC an

RSA while generating public key.

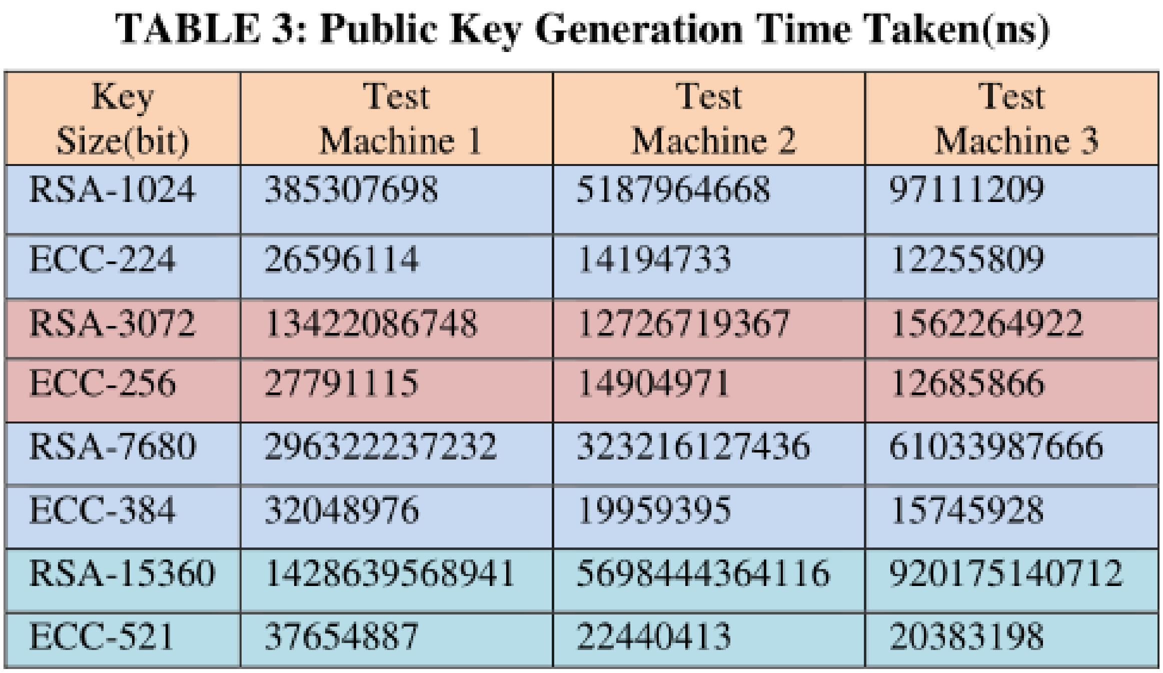


TABLE 3: Public Key Generation Time Taken(ns)

Key Size(bit)	Test Machine 1	Test Machine 2	Test Machine 3
RSA-1024	385307698	5187964668	97111209
ECC-224	26596114	14194733	12255809
RSA-3072	13422086748	12726719367	1562264922
ECC-256	27791115	14904971	12685866
RSA-7680	296322237232	323216127436	61033987666
ECC-384	32048976	19959395	15745928
RSA-15360	1428639568941	5698444364116	920175140712
ECC-521	37654887	22440413	20383198

Figure 7.4: Public Key Generation time taken Table [12]

7.1.5. Encryption and Decryption Time

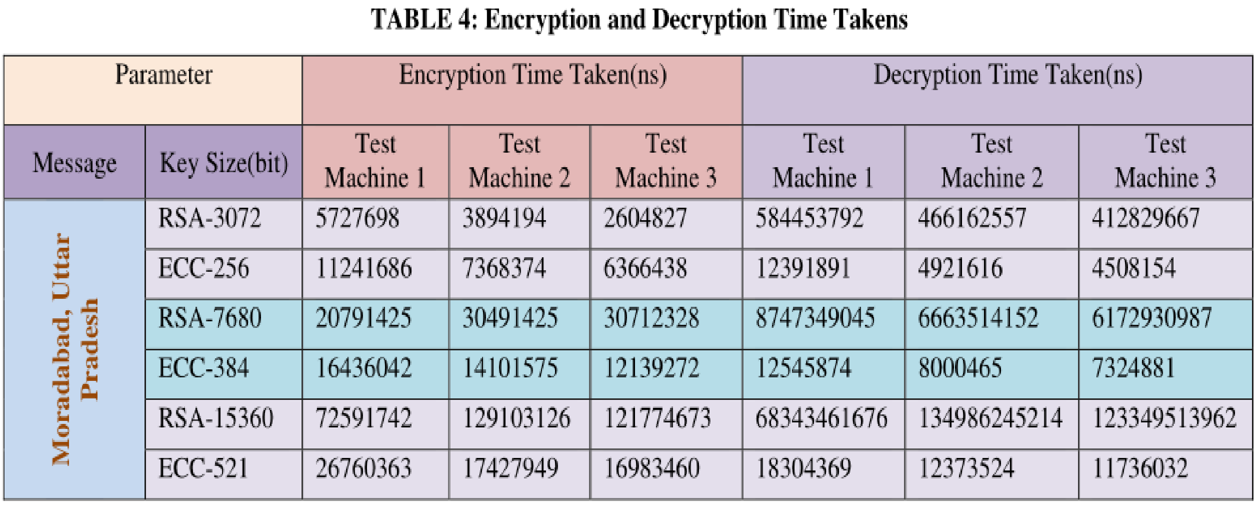


TABLE 4: Encryption and Decryption Time Takens

Parameter		Encryption Time Taken(ns)			Decryption Time Taken(ns)		
Message	Key Size(bit)	Test Machine 1	Test Machine 2	Test Machine 3	Test Machine 1	Test Machine 2	Test Machine 3
Moradabad, Uttar Pradesh	RSA-3072	5727698	3894194	2604827	584453792	466162557	412829667
	ECC-256	11241686	7368374	6366438	12391891	4921616	4508154
	RSA-7680	20791425	30491425	30712328	8747349045	6663514152	6172930987
	ECC-384	16436042	14101575	12139272	12545874	8000465	7324881
	RSA-15360	72591742	129103126	121774673	68343461676	134986245214	123349513962
	ECC-521	26760363	17427949	16983460	18304369	12373524	11736032

Figure 7.5: Table of Encryption and decryption time table[12]

7.1.6. Power Consumption

ECC requires less power for its functioning so it is more suitable for low power applications such as handheld and mobile devices.[12]

7.1.7. Security level(bits) and ratio of cost for RSA and ECC

Key Size		Security Level (bits)	Ratio of cost
RSA	ECC		
1024	160	80	3:1
2048	224	112	6:1
3072	256	128	10:1
7680	384	192	32:1
15360	521	256	64:1

Figure 7.6: Table showing with equivalent Security level and cost ratio[6]

To clarify the comparison, illustrates the data in the above table also which includes cost and security level..

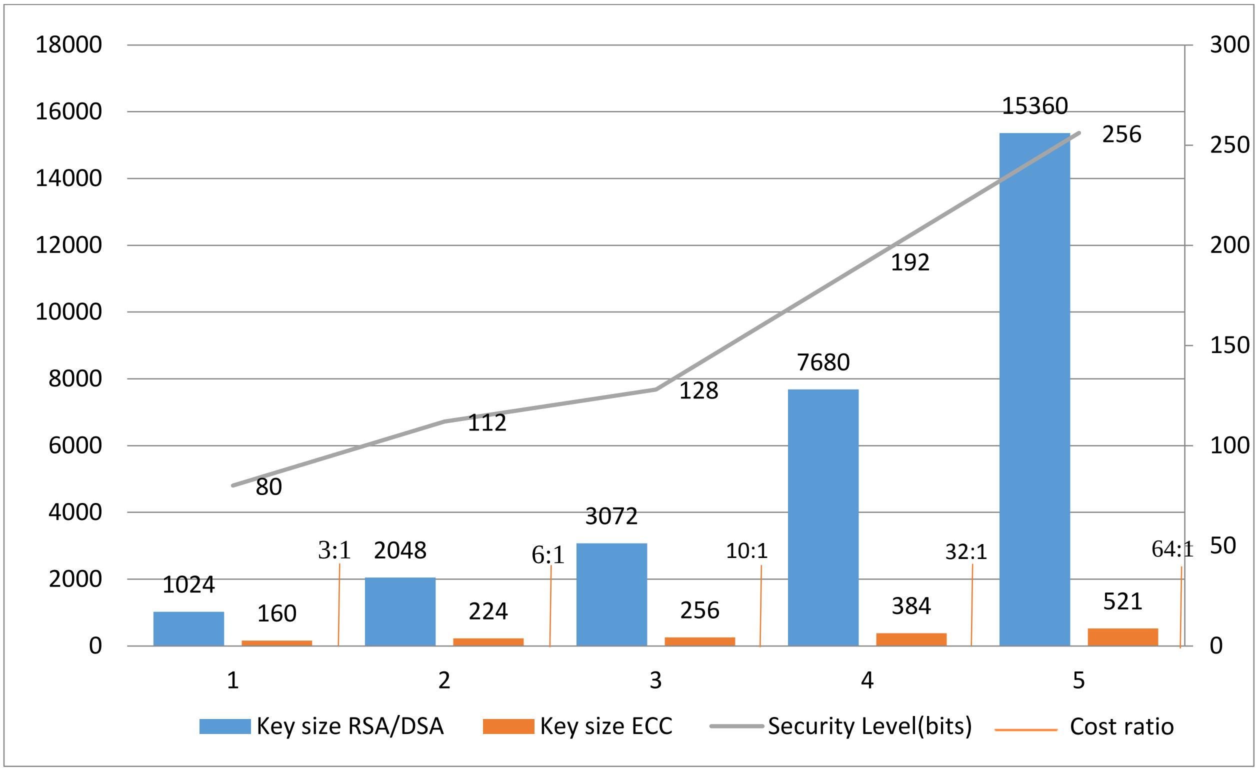


Figure 7.7: Graph of Cost ratio with security level.

7.2 Comparative table :

7.2.1. Comparison between ECC and RSA

Evaluation Criteria	ECC	RSA
1.Memory storage	-Less memory needed Example: Bit size: 160	-More memory needed than ECC. Example: Bit size: 1024
2.Encription Decryption Time	Less encryption and decryption time	More encryption and decryption time
3.Complexity	More complex than RSA	Simpler than ECC
4.Advantages	-hard to break the security.	-Easy to break the security
6.Cost	Less costly than RSA	Costly than ECC
7.Public key generation time	Less time needed	More time needed
8.Power Consumption	Low power consumption	High power consumption than ECC

7.3 Efficiency of threshold cryptography [13]:

- Lagrange interpolation requires O(k log2 k) steps.
- Instead of sharing a singe long s, one can divide s into j smaller pieces and share every piece. Complexity reduces from O(k log2 k) to O(k(log k − log j) 2)
- Size of each share si = size of the secret s

7.3.1 User's (k, n) threshold scheme

Any subset of up to k − 1 shares does not leak any information on the secret

Given k − 1 shares (xi , yi), every candidate secret s '∈ Zp corresponds to an unique polynomial of degree k−1 for which f(0) = s ' . From the construction of polynomials, for all s '∈ Zp, probabilities Pr[s = s ']] are equal. This scheme is perfectly secure and does not depend on the computational power of any party.

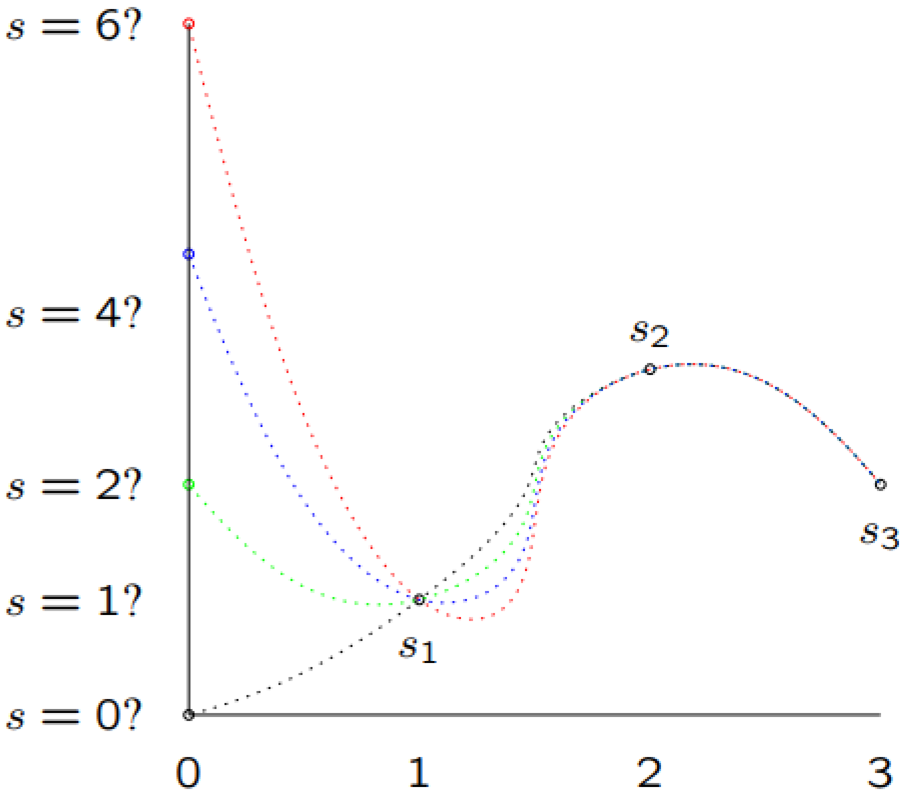


Figure 7.8: Security of threshold scheme illustration

7.4 Recommendation

After analyzing the above data we can come to an conclusion that our proposed model would be a unique one than the existing version of Kerberos authentication protocol in terms of providing essential security, lessening the numbers of bits (memory usages) the system required and also providing more complex structure to those intruder who prove themselves as famous security breaker. Threshold cryptography will ensure more security to the users by introducing multi authentication system. On the other hand, ECC public key cryptography has the ability to ensure the same security as RSA but here it needs less numbers of bits to ensure the same level of security like RSA which ultimately use less memory. Also as ECC is a curve based cryptography and it uses complex calculation to generate keys through which it operates, undoubtedly it is hard to the intruder to break the security key.

7.5 Comparison between our proposed model and existing model

Evaluation Criteria	**Our proposed model**	**Existing model**
1.Algorthms	Threshold and ECC (Combined)	Only RSA or only threshold
2.Memory	Less memory needed as ECC uses less no of bits.	More memory needed
3.Time	More execution time	Less execution time
4.Security level	More secure	Less secure than our model
5.Power consumption	Less power consumption	More power consumption
6. Complexity	More complex	Less complex
7.CPU cycle	Needs less CPU cycle while processing	Needs more CPU cycle while processing

Chapter 8

Conclusion

8.1 Limitation

In our model, we have used threshold cryptography where we need at least k numbers of response among n numbers of TGS servers. Before that there was only one response. So time requires at least k times more in the new version as the response will come from k numbers of TGS server. This is the only limitation which we have in our proposed model.

8.2 Future Work

In order to provide the authentication to the user, using Threshold Cryptography in Kerberos there is a need to work on time complexity also. In future, this work can be extended by using the concept of parallel processing algorithm to enhance the efficiency of the overall system. To incorporate confidentiality and integrity along with the authentication protocol by proper implementation, can be another aspect to explore the perspective of security in cloud computing environment.

8.3 Reach conclusion

In this paper we have discussed about the need of authentication in cloud computing .It is narrative approach of authentication by Kerberos , threshold cryptography and ECC so that encryption technique is more robust that means secure transaction of data , increase memory efficiency, cost efficiency as well as reduce the burden of computation. Data in different states has been discussed along with the techniques which are efficient for encrypting the data in the cloud. There have lot of work done already on security issues and challenges but still there are loop holes. This work of fiction is unique approach because propose scheme minimizes the problem of exchange of key that are generally occurs in the study provided an overview of block cipher, stream cipher and hash function which are used for encrypting the data in the cloud whether it is at rest or in transit.

Chapter 9

References:

1. Bharill S, Lalwani P, Hamsapriya T. A Novel Approach for Enhancing the Authentication Process in Cloud Computing. ELSEVIER, Proc. of Int. Conf. on Advances in Computer Science, AETACS.

2. Amara M, Siad A. Elliptic Curve Cryptography and its applications. InSystems, Signal Processing and their Applications (WOSSPA), 2011 7th International Workshop on 2011 May 9 (pp. 247-250). IEEE.

3. Hojabri M. Innovation in cloud computing: Implementation of Kerberos version5in cloud computing in order to enhance the security issues. InInformation Communication and Embedded Systems (ICICES), 2013 International Conference on 2013 Feb 21 (pp. 452-456). IEEE.

4. Takabi H, Joshi JB, Ahn GJ. Security and privacy challenges in cloud computing environments. IEEE Security & Privacy. 2010 Nov;8(6):24-31.

5. Liu W. Research on cloud computing security problem and strategy. InConsumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on 2012 Apr 21 (pp. 1216-1219). IEEE.

6. National Security Agency., “The Case for Elliptic Curve Cryptography”,

(nsa.gov), [online] Jan 15, 2009, http://www.nsa.gov/business/programs/elliptic_curve.shtml, (Accessed: 20 July 2013)

7. GeeksforGeeks., “RSA Algorithm in Cryptography”, (geeksforgeeks.org) [online] Retrived on: December 5, 2017,

http://www.geeksforgeeks.org/rsa-algorithm-cryptography/

8. DevCentral., “Real Cryptography Has Curves: Making The Case For ECC”, https://devcentral.f5.com/articles/real-cryptography-has-curves-making-the-case-for-Ecc-20832

(Accessed: 03 December 2017)

9. Verma SK, Ojha DB. A discussion on elliptic curve cryptography and its

applications. IJCS International Journal of Computer Science. 2012(2012):9.

10. Snoop MS. Elliptic curve cryptography: An implementation tutorial. Tata Elxsi Ltd. 2007 Jan 5.

11. Singh SR, Khan AK, Singh SR. Performance evaluation of RSA and Elliptic Curve Cryptography. InContemporary Computing and Informatics (IC3I), 2016 2nd International Conference on 2016 Dec 14 (pp. 302-306). IEEE.

12. Malik MY. Efficient implementation of elliptic curve cryptography using low-power digital signal processor. InAdvanced Communication Technology (ICACT), 2010 The 12th International Conference on 2010 Feb 7 (Vol. 2, pp. 1464-1468). IEEE.

13. www.wekipedia.com

14. EposLink., “Off Site Backups” , http://www.eposlink.co.uk/off-site-backups/ (Accessed: 03 December 2017)

16. TechTarget., “Five steps to using the Kerberos protocol” ,http://searchwindowsserver.techtarget.com/feature/Five-steps-to-using-the-Kerberos-protocol