Development of a Multilevel Authentication System for Data Center Administration

Mohammad Taifur Islam Akhand
Student ID: 012163004

A Project

in

The Department

of

Computer Science and Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Science in Computer Science and Engineering

United International University

Dhaka, Bangladesh

March 2019

Approval Certificate

This project titled "Development of a Multilevel Authentication System for Data Center Administration" submitted by **Mohammad Taifur Islam Akhand**, Student ID: 012163004, has been accepted as Satisfactory in fulfillment of the requirement for the degree of Master of Science in Computer Science and Engineering on 23.03.2019.

Board of Examiners

1.

_____ Supervisor

Mohammad Mamun Elahi
Assistant Professor
Department of Computer Science and Engineering
United International University
United City
Madani Avenue
Badda, Dhaka 1212, Bangladesh.

2.

_____ Head Examiner

Dr. A.K.M. Muzahidul Islam
Professor
Department of Computer Science and Engineering
United International University
United City
Madani Avenue
Badda, Dhaka 1212, Bangladesh.

3.

_____ Ex-Officio

Dr. Mohammad Nurul Huda
Professor & Director, MSCSE
Department of Computer Science and Engineering
United International University
United City
Madani Avenue
Badda, Dhaka 1212, Bangladesh.

Declaration

This is to certify that the work entitled "**Development of a Multilevel Authentication System for Data Center Administration** " is the outcome of the research carried out by me under the supervision of Mohammad Mamun Elahi, Assistant Professor.

_____

Mohammad Taifur Islam Akhand
Student ID: 012163004
Department of Computer Science and Engineering
United International University
United City
Madani Avenue
Badda, Dhaka 1212, Bangladesh.

In my capacity as supervisor of the candidate's project, I certify that the above statements are true to the best of my knowledge.

_____

Mohammad Mamun Elahi
Assistant Professor
Department of Computer Science and Engineering
United International University
United City
Madani Avenue
Badda, Dhaka 1212, Bangladesh.

Abstract

Authentication provides access to server, networking equipment, application and other services after verify user's credential from authentication server. Authentication system is important because it keeps organization secure by allowing only authenticated user to access the network resources. Administration of Data Center is another area, where maintaining secured access is very important. Today UNIX administrators use only user name and password to access server for data center administration. Traditional user name and password is vulnerable because attack can happen by password theft, snooping, brute force attack. To protect login information of administrator, we have developed a multilevel authentication system that provides strong three level protection since administrator will be prompted for password, verification code and single sign on. Once three level will be matched with authenticate server, administrator will be allowed to access server for administration in data center. In multilevel authentication system we have used password, Time based One Time Password, single sign on.

Key words: Multi level authentication; data center; TOTP; single sign on; verification code.

Acknowledgement

At first I would like to express my gratitude to almighty Allah for giving me knowledge and strength to complete the project successfully.

This project “Development of a Multilevel Authentication System for Data Center Administration” is prepared for the partial fulfillment of the requirements for the degree of Master of Science in computer science and engineering.

I would like to express my gratitude to my honorable supervisor Mohammad Mamun Elahi, Assistant professor for the useful comments, remarks and engagement of this project.

I would also like to express my gratitude to Dr. A.K.M. Muzahidul Islam, professor for the useful comments, remarks of this project.

I would also like to express my gratitude to Dr. Mohammad Nurul Huda, professor & Director for the useful comments, remarks of this project.

Finally, I wish to thank my parents, wife and son for their support and encouragement throughout my study.

Table of Contents

LIST OF TABLES vii

LIST OF FIGURES viii

1. Introduction 1

1.1 Motivation 2

1.2 Theoretical Background 2

1.3 Problem statement 2

1.4 Objectives 3

1.6 Organization of the report 3

2. Background and Literature Review 4

2.1 Different techniques involved in authentication 4

2.2 Different Techniques Involved in Generation of One Time Password 5

2.2.1 One Time Password (OTP) 5

2.2.2 HMAC based One Time Password (HOTP) 5

2.2.3 Time based One Time Password (TOTP) 5

2.3 Network Information service (NIS) 6

2.4 Related Work 6

3. Proposed Solution 8

3.1 Working Procedure 9

3.2 Application 11

4. Methodology 12

4.1 Sequence diagram 12

4.2 Use case diagram 13

5. Experimental setup and testing 15

5.1 Hardware/Software requirement 15

5.3 Testing 28
6. Results and Discussion 30
7. Conclusions 32
8. References 33

LIST OF TABLES

Table 1: Comparison between TOTP and HOTP 6

Table 2: Hardware/software list 15

LIST OF FIGURES

Figure 1: Network Information Service (NIS).....6

Figure 2: Proposed System Design.....9

Figure 3: Working procedure.....10

Figure 4: Block diagram of multilevel authentication system.....12

Figure 5: Sequence diagram 13

Figure 6: Use case diagram.....14

Figure 7: named .conf 16

Figure 8: forward zone.....16

Figure 9: Reverse zone 17

Figure 10: host file of Kerberos server 18

Figure 11: kdc.conf 18

Figure 12: kadm5.acl 19

Figure 13: krb5.conf 19

Figure 14: listprincs 21

Figure 15: ssh_config 21

Figure 16: hosts file 22

Figure 17: krb5.conf 22

Figure 18: sshd_config 23

Figure 19: ssh_config 23

Figure 20: hosts 24

Figure 21: krb5.conf 24

Figure 22: sshd_config 25

Figure 23: ssh_config 25

Figure 24: google authenticator 26
Figure 25: QR code 26
Figure 26: user profile 26
Figure 27: Restriction the use of same authentication token 26
Figure 28: Rolling window 27
Figure 29: Rate Limits 27
Figure 30: syslog 27
Figure 31: all VMs 28
Figure 32: password 28
Figure 33: prompt for verification code 28
Figure 34: google authenticator 28
Figure 35: Verification code 29
Figure 36: Kerberos user login and ticket generating 29
Figure 37: login krbclient2 by single sign on 29

Chapter 1

Introduction

In the context of computer systems, authentication is a process that ensures and confirms a user's identity [1]. Multilevel authentication provides strong authentication by utilizing password, verification code that will be generate on mobile phone for time based onetime password (TOTP) and single sign on that authenticate with Kerberos server. A time based one time password (TOTP) is a temporary passcode, generated by an algorithm, for use in authenticating access to computer systems [2]. In order to protect administrator's login information we have developed multilevel authentication system. In this system we have used password, verification code and single sign on. In order to get verification code we have used Google authenticator that will generate time based one time password (TOTP) in every 30 seconds. The Google authenticator use time based one time password (TOTP) algorithm. This time based one time password involves OTP from shared secret key and the current system time with a cryptographic hash function. The Google authenticator works offline. Google authenticator use special code or QR code and this QR code contains a shared secret key. The timestamp increase in a fixed interval i.e. 30 seconds in both Google authenticator app and server and both are synchronized at time of set up.

In this multilevel authentication system, we have used single sign on. The single sign on (SSO) provides centralized authentication from authentication server. We have used Kerberos server for centralized authentication. Kerberos server is a strong authentication system for server, client by using secret key cryptography. The illegal users cannot logging to the system through illegal operation due to Kerberos server's identity authentication facilities.

In this multilevel authentication system, we have also used DNS. The domain name system (DNS) "multias.com" has used for name resolution. In order to configure Kerberos server we have used "krbserver.multias.com", "krbclient1.multias.com","krbclient2.multias.com". We have configured forward zone "forward.multias.com" and revers zone "reverse.multias.com" by entering "krbserver.multias.com", "krbclient1.multias.com", "krbclient2.multias.com" for name base resolution.

In this system, we have configure syslog server. The syslog server will collect all login related information to syslog server for investigating user related information during access the server for data centre administration.

1.1 Motivation

A data center is consist of networking equipment, server, storage and other infrastructure related equipment. The data center is center point for banking or organization to store data and access data from data center. The data center is heart of any organization because business related application, services run from data center. A data center administrator does the data center operation for business critical operation, maintenance, infrastructure design and management. Therefore administrator's login information is highly important for an organization. In order to protect administrator's login information from any kinds of unwanted situation, multilevel authentication is required for data center administration.

1.2 Theoretical Background

Most UNIX user or administrator log in to UNIX server from putty. The putty application is running on user's or administrator's PC. During login to the UNIX system, user or administrator use only username and password. This login method is very simple. Traditional username and password-based single-factor authentication is easy to deploy but vulnerable to dictionary attacks, snooping, and brute force attacks [3]. Because if any reason PC is compromised by the hacker by using password theft, man in the middle attack ,sniffing then hacker can easily access to the UNIX server and can steal lots of information. The conventional SMS authentication using a mobile phone is no longer a secure means of authentication. This is because SMS authentication is not multi-channel authentication [4].

1.3 Problem statement

To steal password from computer is simple if network is not secure. In cryptography and computer security, a man-in-the-middle attack (MITM) is an attack where the attacker secretly relays and possibly alters the communication between two parties who believe they are directly communicating with each other [5]. Since multilevel authentication provides 3 level authentication password, time based onetime password (TOTP),single sign on (SSO), we can protect administrator's login information from password theft, sniffing and man in the middle attack by using multilevel authentication during data center activities.

1.4 Objectives

To protect administrator's or sensitive user's login information from any kinds of unwanted situation is the main objective of this project. In multilevel authentication system password, time based onetime password (TOTP) and single sign on provides strong authentication for data centre administrator. Data centre administrator will be protected from password theft, man in the middle attack and sniffing by using multilevel authentication.

1.6 Organization of the report

The organization of the report as follows, chapter 2 contains background and literature review, chapter 3 contains proposed solution, chapter 4 contains methodology, chapter 5 contains experimental setup and testing, chapter 6 contains results and discussion, chapter 7 contains conclusions.

Chapter 2

Background and Literature Review

Most of UNIX user or administrator log in to UNIX server from putty. The putty application is running on user's or administrator's PC. During login to the UNIX system, user or administrator use only username and password. This login method is very simple. Traditional username and password-based single-factor authentication is easy to deploy but vulnerable to dictionary attacks, snooping, and brute force attacks [3]. The poor security practices at the server's leads to stolen password files that are easily compromised using an offline attack and passwords are too easily stolen via phishing attacks [6].

The conventional SMS authentication using a mobile phone is no longer a secure means of authentication. This is because SMS authentication is not multi-channel authentication [4]. Phishing, a serious security threat to Internet users is an e-mail fraud in which the perpetrator sends out an email which looks like legitimate, in an order to gather personal and financial information of the receiver [7].

2.1 Different techniques involved in authentication

2.1.1 SMS based authentication

SMS is a store-and-forward messaging system for cell phones and SMS OTP traverses multiple hops across carriers. It has some drawback,

i It becomes vulnerable due to network congestion.

ii If operator service outages in any reasons then gateway become down. This gateway down time also affects SMS-based OTPs.

iii The gateway compromised is a major security breach, especially when SMS traverse overseas gateways.

iv SIM cloning is another threat for SMS-based OTP.

2.1.2 Email based authentication

In email based authentication, we get a code which uses for next step authentication. But email based authentication is most vulnerable. Because during its travel through network, it can be hacked by man in the middle attack or sniffing.

2.2 Different Techniques Involved in Generation of One Time Password

2.2.1 One Time Password (OTP)

OTP means one time password is a set of characters which act as a kind of identity for just once. Once it is used it will not use for any further authentication.

One time password can be generated in any of the two ways:

Time-synchronized OTP

In time-synchronized OTPs the user should enter the password within a certain period of time else it gets expired and another OTP must be generated.

Counter-synchronized OTP

In counter-synchronized OTPs, a counter is synchronized between the client device and the server. The device counter is advanced each time an OTP is requested.

There are two main standard for one time password i.e. HOTP, TOTP

2.2.2 HMAC based One Time Password (HOTP)

HOTP means HMAC based One Time Password. It has two pieces of information. The first is secret key called “seed” and the second piece of information is moving factor is called counter. The counter is stored in the token and server. The counter in token is incremented when button on the token is pressed, while the counter on the server is incremented when an OTP is successfully validated.

2.2.3 Time based One Time Password (TOTP)

TOTP means Time based One Time Password. It is based on HOTP but moving factor based on time instead of counter. It uses time in increments called the timestamp, which is usually

30 or 60 seconds. This means that each one time password (OTP) is valid for the duration of the timestamp.

There are some comparison between TOTP and HOTP as shown in Table 1.

Table 1: Comparison between TOTP and HOTP

TOTP	HOTP
TOTP stands for Time based One Time Password	HOTP stands for HMAC based One Time Password
TOTP passwords are short-lived	HOTP passwords are potentially longer lived
They only apply for a given amount of human time	They apply for an unknown amount of human time
TOTP changes in every 30 or 60 second	HOTP stays for long time
TOTP increase with timestamp	HOTP increase with counter increase

2.3 Network Information service (NIS)

In Network Information Service (NIS) authentication server, there is no secure propagation of user authenticators as shown in Figure 1. If anyone using a sniffer is able to get all the clear text or hashed password propagated on the network.

If NIS client use broadcast service to locate NIS server on the network, the intruders can easily introduce their own NIS server by their own account. Once a client bind to the rogue NIS server, the intruders can do the unauthorized access.

If NIS is used for authentication it send password over the network in clear text can easily captured and make the system vulnerable.

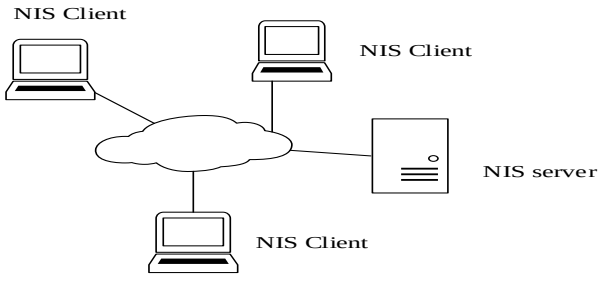

Figure 1: Network Information Service (NIS)

2.4 Related Work

Fadi Aloul, Syed Zahidi, Wassim El-Hajj proposed a 2 factor authentication using mobile phone [8]. The system consists of a server connected to a GSM modem and a mobile phone client running a J2ME application. They have used mobile phone to generate OTP as

software token and OTP is valid for short period of time and are unique for both user and mobile phone device itself. Proposed SMS-based authentication.

Eko Sediyono, Kartika Imam Santoso, Suhartono have done research on Secure Login by Using One-time Password Authentication Based on MD5 Hash Encrypted SMS[9]. They have used PHP programming, MySQL and Gammu SMS gateway server to send OTP via SMS to authenticate user. They have developed authenticated and non-memorized OTP.

In this project we have developed multilevel authentication system password, verification code, and single sign on. The administrator user will first use password then verification code that will be generated on Mobile by Google authenticator app. This Google authenticator use time based one time password (TOTP) algorithm that use timestamp. The timestamp of server and mobile phone have to synchronize. The google-authenticator service run on server and generate time based token and same time one time password (OTP) generate on mobile. For time synching with mobile and sever, no network connection is required. We have used Kerberos system for single sign on. Kerberos is network authentication protocol that allow nodes communicating over a network to prove identity to one another in a secure manner.

The developed multilevel authentication system is secure for authentication.

Chapter 3

Proposed Solution

The proposed solution consists of: DNS server, Kerberos server, Kerberos client, Google authenticator for TOTP, syslog server. The proposed system diagram is shown in Figure 2.

DNS server

The DNS is name resolution server. The domain name system (DNS) “multias.com” has used for name resolution. In order to configure Kerberos server we have used “krbserver.multias.com”, “krbclient1.multias.com”,”krbclient2.multias.com”. We have configured forward zone “forward.multias.com” and revers zone “reverse.multias.com” by entering “krbserver.multias.com”, “krbclient1.multias.com”, ”krbclient2.multias.com” for name base resolution.

Kerberos server

Kerberos server provide strong authentication for server, client applications by using secret key cryptography. The Kerberos server provide identity authentication that can prevent illegal users from logging on the system to obtain improper benefit through illegal operation. We have used single sign on. The single sign on (SSO) provides centralized authentication from authentication server.

Kerberos client

The Kerberos client will be authenticate with Kerberos server to access the system.

Google authenticator for TOTP

Google authenticator use special code or QR code and this QR code contains a shared secret key. The timestamp increase in a fixed interval i.e. 30 seconds in both Google authenticator app and server and both are synchronized at time of set up.

Syslog server

The syslog server will collect administrator’s login related information in syslog server.

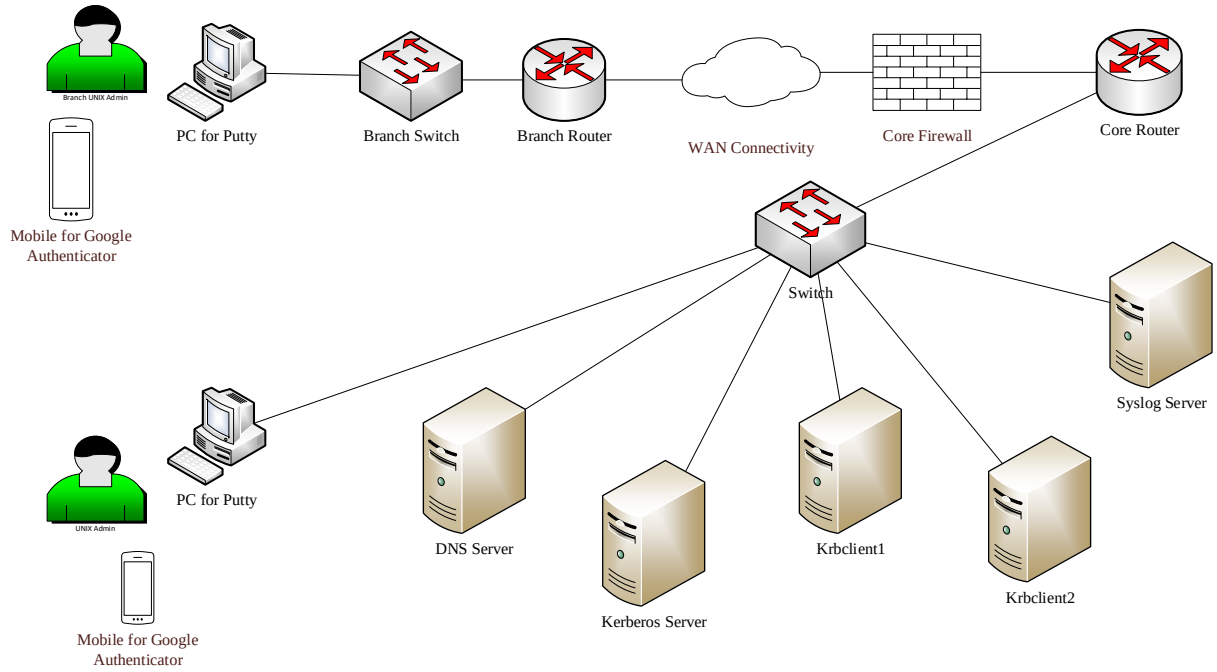

Figure 2: Proposed System Design

3.1 Working Procedure

The UNIX Admin will connect to Kerberos client from their PC through Putty. During connect from putty they will be asked for user name, password. After giving username, password, they will be asked for verification code. UNIX admin will get 6 digit TOTP from Google authenticator from their mobile. Once matched verification code, UNIX admin will be allowed for successful login to Kerberos client. The UNIX admin will use “su” to Kerberos user. After giving password Kerberos user will be authenticate from Kerberos server. UNIX admin will run “klist” to get ticket for 1 minute time period. UNIX admin will access to main sever by single sign on. The detail working procedure of proposed system as shown in Figure 3.

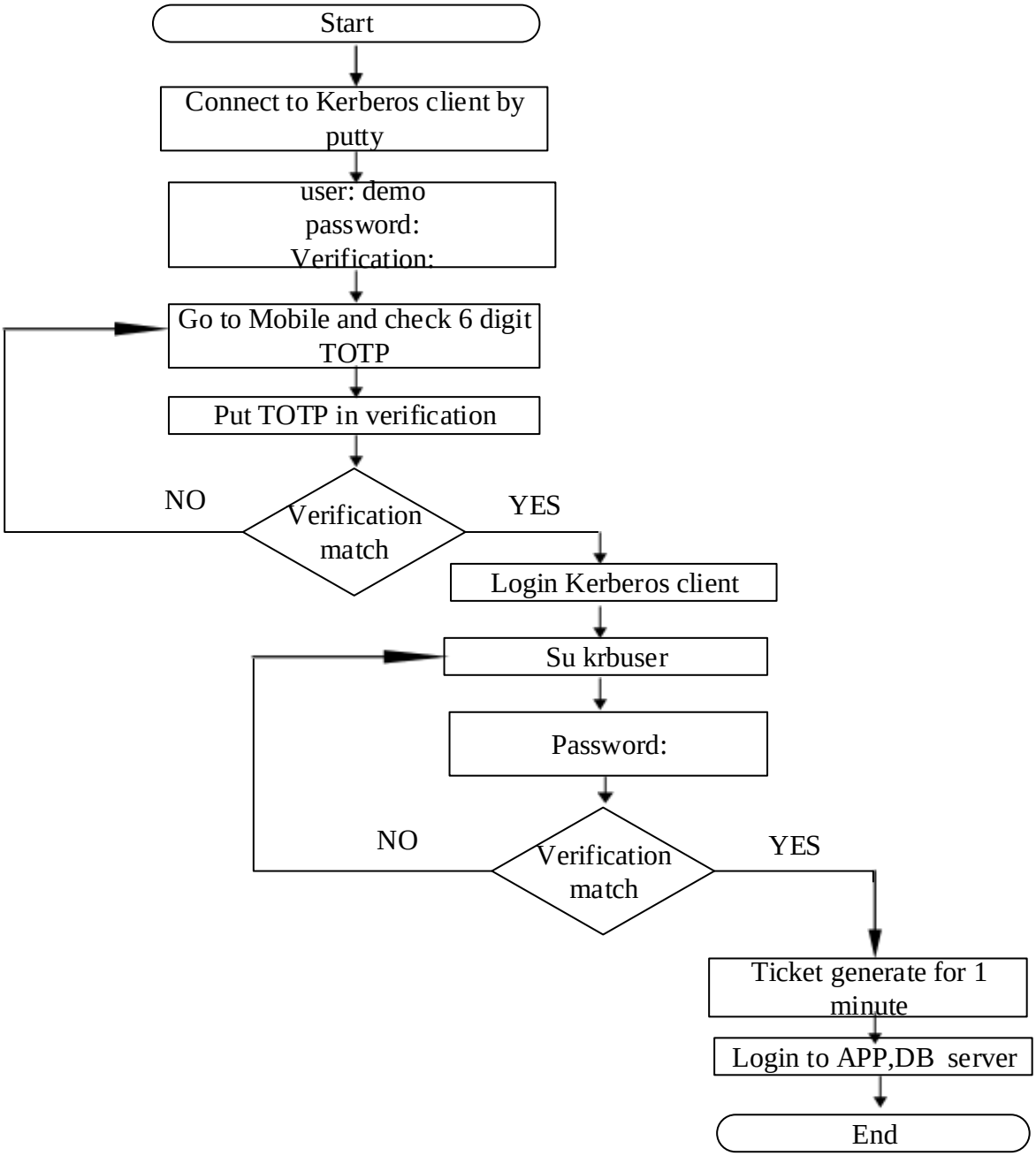

Figure 3: Working procedure

3.2 Application

Every organization has data center for running IT equipment server, storage and infrastructure related equipment. We can deploy our proposed solution in following organization,

1. University

Every university has data center. In data center many servers are running for various application like student management system, salary software, HRM software. By using multilevel authentication system in data center, administrator will be secured to do their day to day activities.

2. Banking Sector

Banks are most important wing of any country. A bank can provide valuable service to a country. A bank plays a vital role in economic matter of country. At present every banking sector has data center to run various server, storage, networking equipment to run banking services. So, it is very important to protect banking authentication system. By using multilevel authentication system, we can protect banking sector from any kind unauthenticated situation like password theft, man in the middle attack, sniffing. The data center administration can do the activity by using multilevel authentication system.

Chapter 4

Methodology

At first user or administrator will connect to the Kerberos client by putty from PC using ssh. When user will connect to the Kerberos client it will prompt for user name and password. After giving user name and password it will prompt for verification code. The user will get 6 digit code from Google authenticator app for verification code. When verification code will be matched with server, it will allow to login the Kerberos client. After that user will use Kerberos user to authenticate with Kerberos server. User will use 'klist' to get ticket. User will login to the server through single sign on. The block diagram and working procedure of multilevel authentication system as shown in Figure 4.

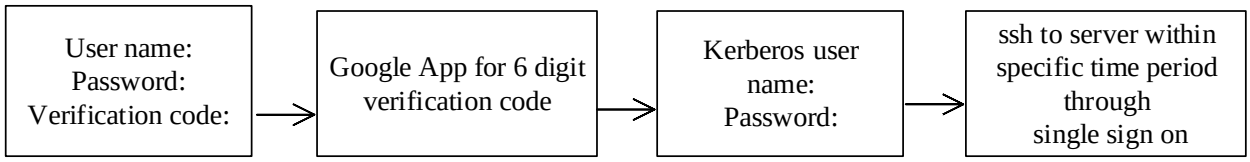

Figure 4: Block diagram of multilevel authentication system

4.1 Sequence diagram

The sequence diagram is illustrated in Figure 5 regarding multilevel authentication for UNIX admin as follows,

User will connect to Kerberos client, it will prompt for user name, password and verification code. After giving user name, password user will check verification code from Google authenticator app for TOTP. Once user name, password, verification code complete, it will allow to the Kerberos client. After that user will use Kerberos user to authenticate with the Kerberos server. User will run 'klist' to get ticket for 1 minute. After that User will connect to server through single sign on within 1 minute.

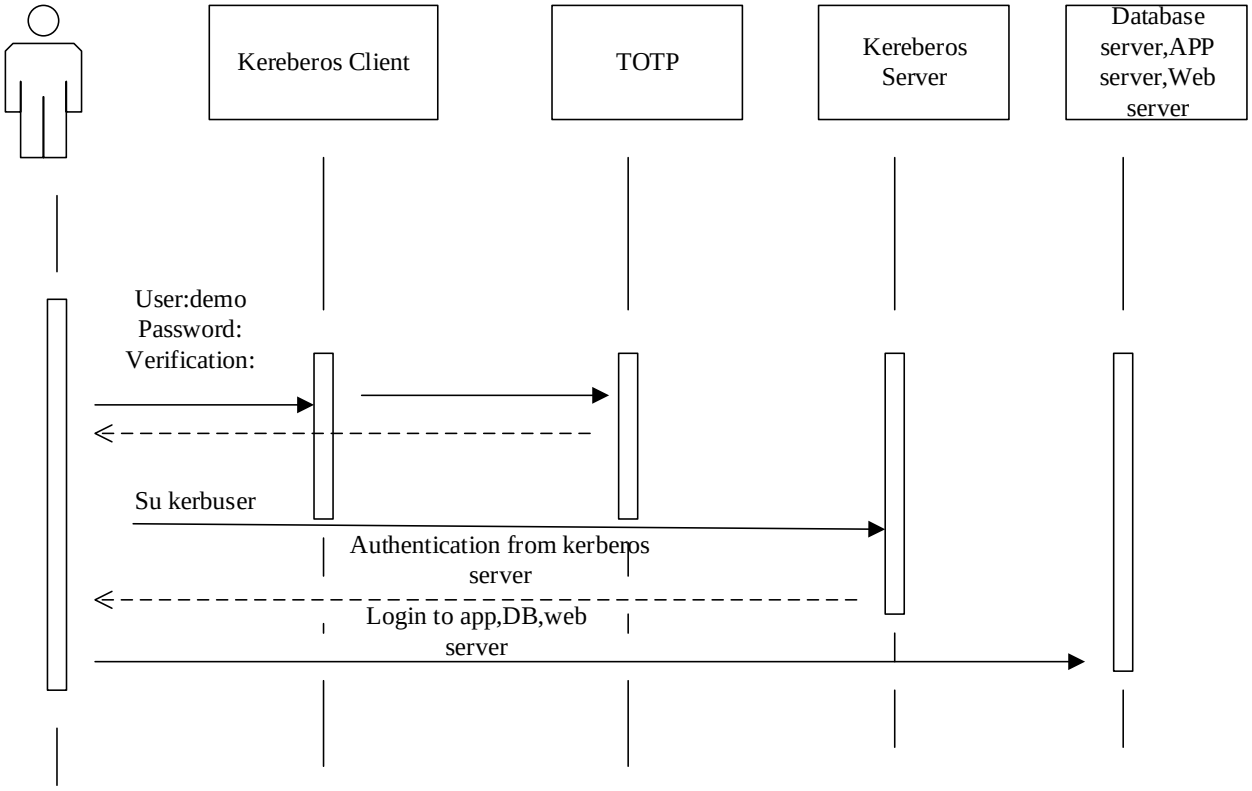

Figure 5: Sequence diagram

4.2 Use case diagram

The use case diagram is illustrated in Figure 6 regarding multilevel authentication for UNIX admin as follows

Step 1: UNIX User will connect to Kerberos client.

Step 2: UNIX User will connect to Kerberos client.

Step 3: It will prompt for user name, password and verification code.

Step 4: After giving user name, password user will get TOTP for verification code from Google authenticator app.

Step 5: When verification code match, user will be successfully login to Kerberos client.

Step 6: User will use ‘su’ to Kerberos user. Authenticate from Kerberos server.

Step 7: User will run ‘klist’ to get ticket for 1 minute.

Step 8: User will connect to server through single sign on.

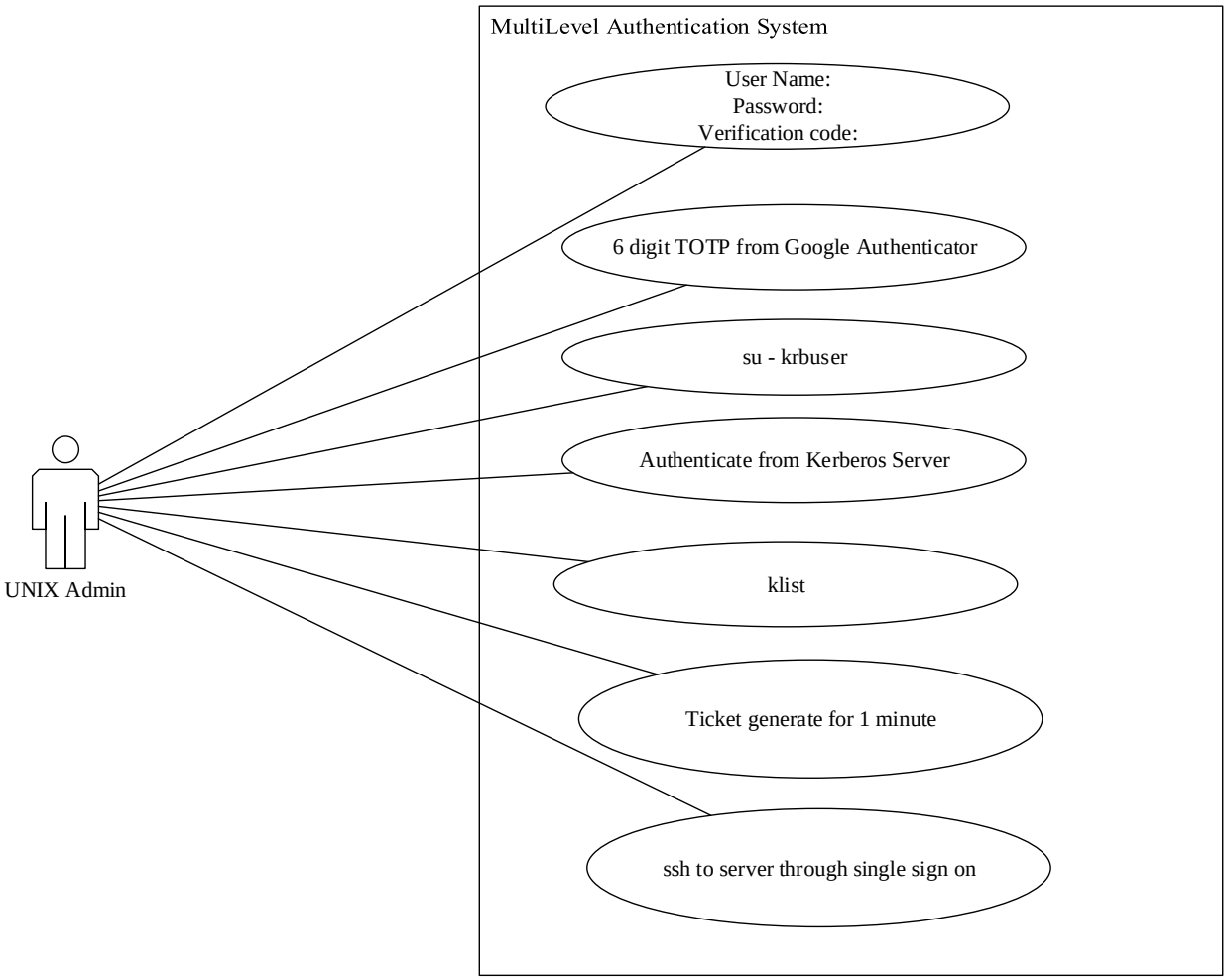

Figure 6: Use case diagram

Chapter 5

Experimental setup and testing

5.1 Hardware/Software requirement

To develop a multilevel authentication system, we have used 5 VMware, Linux CentOS7 64 bit operating system, Google authenticator, smart mobile. The configuration of VMware as shown in Table 2.

Table 2: Hardware/software list

Host Name	Operating system	VMware details	Purpose
Master	Linux Centos7 64 bit	Memory:512MB Processor:1 Disk size=40GB	Primary DNS server
Kerberos Server	Linux Centos7 64 bit	Memory:512MB Processor:1 Disk size=50GB	Kerberos Server
Krbclient1	Linux Centos7 64 bit	Memory:512MB Processor:1 Disk size=50GB	Kerberos client
Krbclient2	Linux Centos7 64 bit	Memory:512MB Processor:1 Disk size=50GB	Kerberos client
Syslog server	Linux Centos7 64 bit	Memory:512MB Processor:1 Disk size=50GB	Syslog server

5.2 Implementation

DNS

DNS is a protocol to exchange data on the internet and on many private networks for computers. It's job is to turn a user friendly domain name into internet protocol (IP) address that computers use to identify each other on the internet [10].

In this project "Multilevel authentication system for data center administration", we will configure a DNS first. To configure a DNS we need following rpm packages for CentOS7 Linux environment i.e. bind 9.9.4-37.el7, bind-utils-9.9.4-37.el7. After installing these packages we will configure named.conf, forward zone (forward.multias.com) and reverse zone (reverse.multias.com) as shown in Figure 7, 8, 9 respectively.

[root@master ~]# cat /etc/named.conf

```
zone "." IN {
        type hint;
        file "named.ca";
};

zone "multias.com" IN {
type master;
file "forward.multias.com";
allow-update {none; };
};

zone "0.168.192.in-addr.arpa" IN {
type master;
file "reverse.multias.com";
allow-update {none; } ;
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

Figure 7: named .conf

[root@master ~]# cat /var/named/forward.multias.com

```
$TTL 1D
@       IN SOA  master.multias.com. root.multias.com.   (
                                        0       ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum

@       IN      NS      master.multias.com.
@       IN      A       192.168.0.205
master  IN      A       192.168.0.205
host    IN      A       192.168.0.205
krbserver IN    A       192.168.0.206
krbclient1 IN   A        192.168.0.207
krbclient2 IN   A       192.168.0.208
syslogserver IN A       192.168.0.209
```

Figure 8: forward zone

[root@master ~]# cat /var/named/reverse.multias.com

```
$TTL 1D
@       IN SOA  master.multias.com. root.multias.com.   (
                                        0       ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum

@       IN      NS      master.multias.com.
@       IN      PTR     multias.com.
master  IN      A       192.168.0.205
host    IN      A       192.168.0.205
krbserver IN    A       192.168.0.206
krbclient1 IN   A       192.168.0.207
krbclient2 IN   A       192.168.0.208
syslogserver IN A       192.168.0.209
205     IN      PTR     master.multias.com.
206     IN      PTR     krbserver.multias.com.
207     IN      PTR     krbclient1.multias.com.
208     IN      PTR     krbclient2.multias.com.
209     IN      PTR     syslogserver.multias.com.
```

Figure 9: Reverse zone

Kerberos

Kerberos is a computer network authentication protocol that works on the basis of tickets to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner [11].

In Kerberos, a client (user or a service) sends a request to the Key Distribution Center (KDC) for a ticket. The Key Distribution Center generate a ticket granting ticket (TGT) and encrypts it. Then client (user or service) tries to decrypt the ticket granting ticket (TGT) by using password. If the ticket granting ticket (TGT) successfully decrypt (i.e. by correct password) by the client, it keeps the decrypted ticket granting ticket (TGT), which indicates proof of the client's identity. The ticket granting ticket (TGT) which expires at a specified time and permits the client to obtain additional tickets, which give permission for specific services. The requesting and granting of these additional tickets is user-transparent.

Kerberos server setup

In this project we have used “Krb5-server, krb5-workstation, pam_krb5” packages for configuring Kerberos server and Kerberos client.

To configure Kerberos server we have added IP address and hostname of Kerberos server, Kerberos client to **/etc/host** file as shown in Figure 10.

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.0.206 krbserver.multias.com
192.168.0.207 krbclient1.multias.com
192.168.0.208 krbclient2.multias.com
```

Figure 10: host file of Kerberos server

Now we will add the DNS name "multias.com" in kdc.conf which reside at /var/Kerberos/krb5kdc/kdc.conf as shown in Figure 11.

[root@krbserver ~]# vi /var/kerberose/krb5kdc/kdc.conf

```
[kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

[realms]
 MULTIAS.COM = {
  master_key_type = aes256-cts
  default_principle_flags= +preauth
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
  supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal
des-cbc-md5:normal des-cbc-crc:normal
 }
```

Figure 11: kdc.conf

The kadmin deamon uses an access control list to manage access rights to the Kerberos database. The location of kadm5.acl at /var/kerberos/krb5kdc/kadm5.acl as shown in Figure 12. In this Figure any principal in the MULTIAS.COM realm with an admin instance has all administrative privilege.

kadmind starts the Kerberos administration server and it runs on the master Kerberos server, which stores the KDC database. The function of kadmind to accept the remote request from kadmin and kpasswd to administer the information in these database.

[root@krbserver ~]# vi /var/kerberos/krb5kdc/kadm5.acl

Figure 12: kadm5.acl

Now we have configured the KDC namely krb5.conf which resides in /etc/krb5.conf as shown in Figure 13. We have configured "ticket_lifetime" 1 minute for security purpose. The life time of ticket granting ticket (TGT) is 1 minute and expires after one minute.

[root@krbserver ~]# vi /etc/krb5.conf

Figure 13: krb5.conf

Now, we will create user in OS level.

[root@krbserver ~]# useradd krbtest

After that we have created internal database. The internal database contains all principals and their passwords. We will be prompted for master password during creation of internal database. This master password is the main key that encrypt all principals in it's database. This password will be used for encryption of database and low level maintenance.

[root@krbserver ~]# kdb5_util create -s -r MULTIAS.COM

Now we have enabled and started krb5kdc, kadmin deamon as follows:

[root@krbserver ~]# systemctl enable krb5kdc

[root@krbserver ~]#systemctl enable kadmin

[root@krbserver ~]#systemctl start krb5kdc

[root@krbserver ~]# systemctl start kadmin

Now , we have configured firewalld to accept Kerberos related traffic:

[root@krbserver ~]# systemctl start firewalld.service

[root@krbserver ~]#firewall-cmd --add-service=Kerberos –permanent

[root@krbserver ~]#firewall-cmd --add-service=kadmin –permanent

[root@krbserver ~]#systemctl restart firewalld.service

[root@krbserver ~]#systemctl enable firewalld.service

After that we will check the contents of database by using listprinc command. Now we will add “addprinc root/admin” for admin user and password of krbtest for user that user will be used for Kerberos user as follows.

[root@krbserver ~]#kadmin.local

Authenticating as principal root/admin@MULTIAS.COM with password

Kadmin.local:

Kadmin.local:addprinc root/admin

Kadmin.local:addprinc krbtest

Kadmin.local: addprinc –randkey host/krbserver.example.com

Kadmin.local: ktadd host/krbserver.example.com

After that we will check the principal status by using listprinc from kadmin.local as shown in Figure 14.

Figure 14: listprincs

Now we need to configure ssh

[root@krbserver ~]# more /etc/ssh/sshd_config

[root@krbserver ~]# systemctl restart sshd

Now ssh deamon will attempt to authenticate using Kerberos. We need to to configure the ssh client side for Kerberos based authentication as shown in Figure 15. We need to configure the following.

[root@krbserver ~]# more /etc/ssh/ssh_config

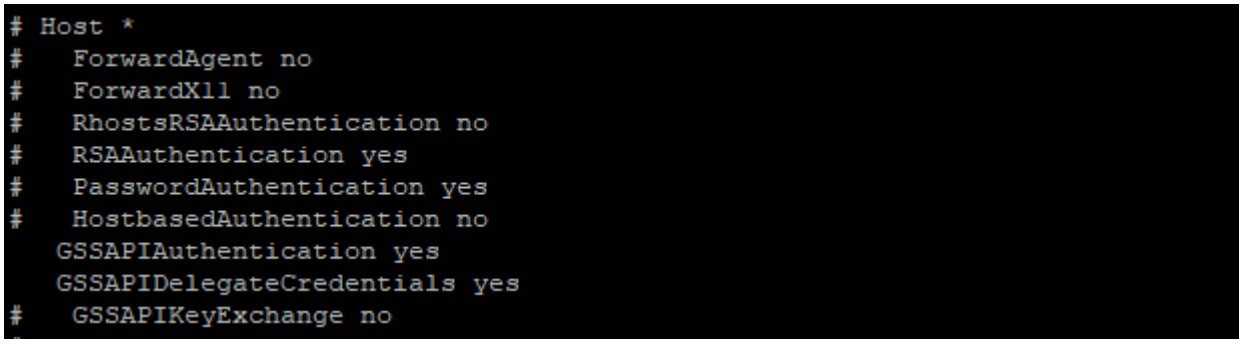

Figure 15: ssh_config

At last we have run following command for updating config file.

[root@krbserver ~]#authconfig –enablekrb5 --update

Kerberos client1 setup

Now we will add IP address and host name of Kerberos's server, Kerberos client1, Kerberos client2 to hosts file which resides in /etc/hosts as shown in Figure 16. After that we will install Krb5-workstation, Pam_krb5 in Kerberos client1.

[root@krbclient1 ~]# vi /etc/hosts

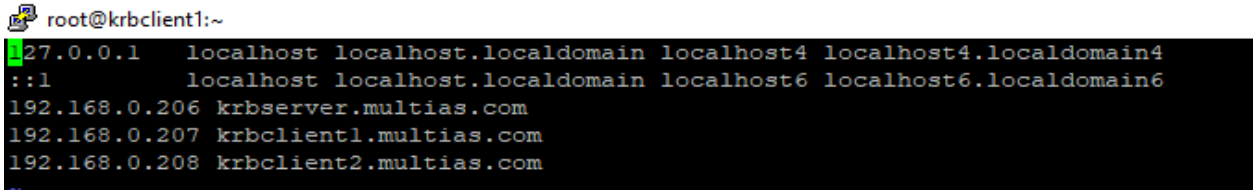

Figure 16: hosts file

Now we will copy krb5.conf from krbserver to krbclient1 to make it identical to krbserver for initiating an authentication KDC server from Kerberos client1 as shown in Figure 17.

[root@krbclient1 ~]# more /etc/krb5.conf

Figure 17: krb5.conf

Now we will run kadmin to create needed principal. The randkey option will ask to generate a random key for this principal.

[root@krbclient1 ~]# kadmin

Authenticating as principal root/admin@MULTIAS.COM with password for root/admin@MULTIAS.COM

Now need to register krbclient1 to krbserver

Kadmin: addprinc -randkey host/krbclient1.multias.com

Kadmin: ktadd host/krbclient1.multias.com

Now we will configure ssh in Kerberos client1 as shown in Figure 18 and Figure 19 respectively,

System security is a crucial constraint for secure transfer of data. Because of the nature of the SSH protocol, anyone with access to the central server can manipulate files, it is imperative that only authorized users be able to access the central server [12].

[root@krbclient1 ~]# more /etc/ssh/sshd_config

Figure 18: sshd_config

[root@krbclient1 ~]# systemctl restart sshd

[root@krbclient1 ~]# more /etc/ssh/ssh_config

Figure 19: ssh_config

[root@krbclient1 ~]# authconfig –enablekrb5 --update

Kerberos client2 setup

Now we will add IP address and host name of Kerbero's servrer, Kerberos client1, Kerberos client2 to hosts file which resides in /etc/hosts as shown in Figure 20. After that we will install Krb5-workstation, Pam_krb5 in Kerberos client2.

[root@krbclient2 ~]#cat /etc/hosts

```
[root@krbclient2 ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.0.206 krbserver.multias.com
192.168.0.207 krbclient1.multias.com
192.168.0.208 krbclient2.multias.com
```

Figure 20: hosts

Now we will copy krb5.conf from krbserver to krbclient1 to make it identical to krbserver for initiating an authentication KDC server from Kerberos client1 as shown in Figure 21.

[root@krbclient2 ~]# more /etc/krb5.conf

```
# Configuration snippets may be placed in this directory as well
includedir /etc/krb5.conf.d/

[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 dns_lookup_realm = false
 ticket_lifetime = 1m
 renew_lifetime = 7d
 forwardable = true
 rdns = false
 pkinit_anchors = /etc/pki/tls/certs/ca-bundle.crt
 default_realm = MULTIAS.COM
 default_ccache_name = KEYRING:persistent:%{uid}

[realms]
MULTIAS.COM = {
  kdc = krbserver.multias.com
  admin_server = krbserver.multias.com
 }

[domain_realm]
 .multias.com = MULTIAS.COM
  multias.com = MULTIAS.COM
```

Figure 21: krb5.conf

Now we will run kadmin to create needed principal. The randkey option will ask to generate a random key for this principal.

[root@krbclient2 ~]# kadmin

Authenticating as principal with password for root/admin@MULTIAS.COM

Now need to register krbclient1 to krbserver

Kadmin: addprinc -randkey host/krbclient2.multias.com

Kadmin: ktadd host/krbclient1.multias.com

Now we will configure ssh to set 'Yes' for GSSAPIAuthentication as shown in Figure 22
[root@krbclient2 ~]# more /etc/ssh/sshd_config

Figure 22: sshd_config

[root@krbclient2 ~]# sytemctl restart sshd

Now we will configure ssh_config to set 'Yes' for GSSAPIAuthentication and GSSAPIDelegatecredentials as shown in Figure 23.

[root@krbclient2 ~]# more /etc/ssh/ssh_config

Figure 23: ssh_config

[root@krbclient2 ~]# authconfig –enablekrb5 --update

Google Authenticator Setup

Google Authenticator is the application based on two-factor Authentication (2FA) that helps for identifying user identity and the confirmation [13].

A pluggable authentication module (PAM) is an application programming interface (API) for authentication related services, which permits system administrators to add new authentication methods by installing PAMs and modifying authentication policies by editing the configuration files [14].

Now we will install rpm of epel-release-latest-7.noarchm PAM, google-authenticator in kerberosclient1.

Now we will run google-authenticator to initialization app as shown in Figure 24.

$google-authenticator

```
[demo@krbclient1 ~]$ google-authenticator

Do you want authentication tokens to be time-based (y/n)
```

Figure 24: google authenticator

This PAM allows for time based or sequential based tokens. The code changes randomly after a certain time elapses. At this point, we will use authenticator app on smart phone to scan the QR code or manually as show type in the secret key as shown in Figure 25.

Figure 25: QR code

After pressing ‘y’ it will prompt for updating demo user’s profile as shown in Figure 26.

```
Do you want me to update your "/home/demo/.google_authenticator" file? (y/n) y
```

Figure 26: user profile

We will prevent multiple uses of same authentication token by pressing ‘yes’. This will help to prevents man in the middle attack as shown in Figure 27.

```
Do you want to disallow multiple uses of the same authentication
token? This restricts you to one login about every 30s, but it increases
your chances to notice or even prevent man-in-the-middle attacks (y/n) y
```

Figure 27: Restriction the use of same authentication token

We will get 2 valid code in 1 minute by answering ‘no’ in rolling window and answering no is the more secure choice as shown in Figure 28.

```
By default, a new token is generated every 30 seconds by the mobile app.
In order to compensate for possible time-skew between the client and the server,
we allow an extra token before and after the current time. This allows for a
time skew of up to 30 seconds between authentication server and client. If you
experience problems with poor time synchronization, you can increase the window
from its default size of 3 permitted codes (one previous code, the current
code, the next code) to 17 permitted codes (the 8 previous codes, the current
code, and the 8 next codes). This will permit for a time skew of up to 4 minutes
between client and server.
Do you want to do so? (y/n) n
```

Figure 28: Rolling window

We have set rate limit for protecting brute-force attack as shown in Figure 29.

```
If the computer that you are logging into isn't hardened against brute-force
login attempts, you can enable rate-limiting for the authentication module.
By default, this limits attackers to no more than 3 login attempts every 30s.
Do you want to enable rate-limiting? (y/n) y
```

Figure 29: Rate Limits

After finishing this setup, we will take back up of secret key. We will also copy the ~/.google-authenticator file to a trusted location. From there, we will deploy it on additional systems or redeploy it after a backup.

Syslog

In computing, syslog is a standard for message logging. It allows separation of the software that generates messages, the system that stores them, and the software that reports and analyzes them [8].

We have configured syslog server that will collect log from syslog client. In this project we have logged in to krbclient1 by using ‘demo’ and all of it’s log has been collected by syslog as shown in Figure 30.

```
Mar 26 12:11:54 krbclient1 systemd: Started Session 2 of user demo.
Mar 26 12:11:54 krbclient1 systemd-logind: New session 2 of user demo.
Mar 26 12:11:54 krbclient1 systemd: Starting Session 2 of user demo.
Mar 26 12:11:54 krbclient1 dbus[697]: [system] Activating service name='org.freedesktop.problems' (using
Mar 26 12:11:55 krbclient1 dbus[697]: [system] Successfully activated service 'org.freedesktop.problems'
Mar 26 12:12:02 krbclient1 dbus[697]: [system] Activating via systemd: service name='net.reactivated.Fpri
Mar 26 12:12:03 krbclient1 systemd: Starting Fingerprint Authentication Daemon...
Mar 26 12:12:03 krbclient1 dbus[697]: [system] Successfully activated service 'net.reactivated.Fprint'
Mar 26 12:12:03 krbclient1 journal: D-Bus service launched with name: net.reactivated.Fprint
Mar 26 12:12:03 krbclient1 fprintd: Launching FprintObject
Mar 26 12:12:03 krbclient1 journal: entering main loop
Mar 26 12:12:03 krbclient1 systemd: Started Fingerprint Authentication Daemon.
Mar 26 12:12:07 krbclient1 su: (to krbtest) demo on pts/0
```

Figure 30: syslog

5.3 Testing

At first we power on DNS_server , krbserver,krbclient1,krbclient2,syslogserver. After powering on VM , we will first login by “demo”user and it will prompt for password and giving password, it will ask for verification in kerberos client1 as shown in Figure 31, 32, 33, respectively.

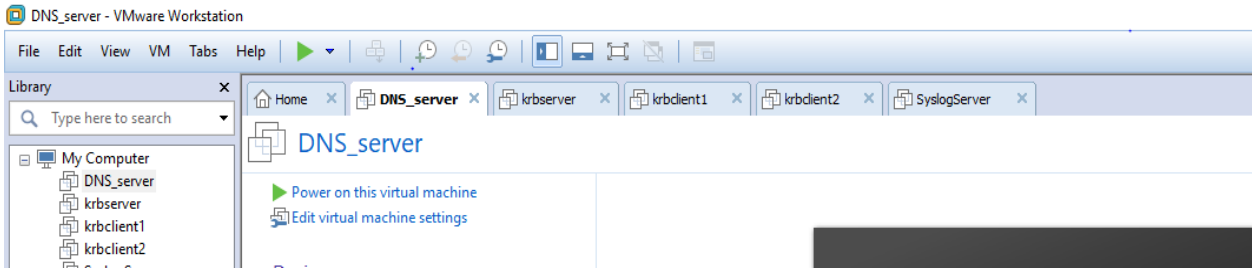

Figure 31: all VMs

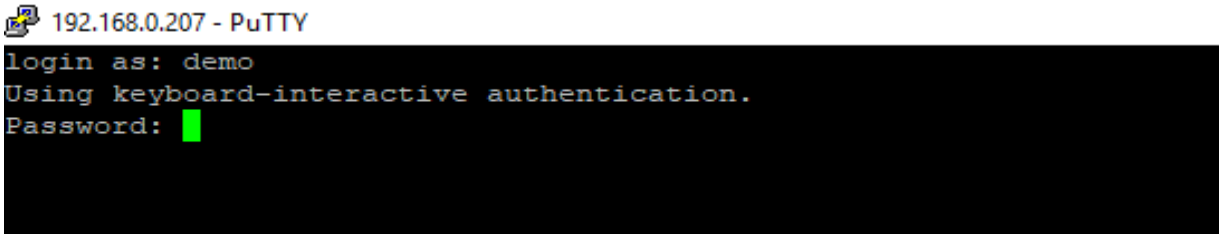

Figure 32: password

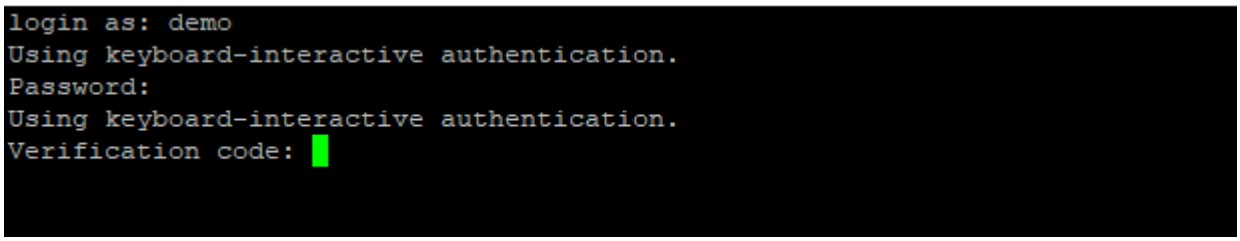

Figure 33: prompt for verification code

After that we will check Time based one time password (TOTP) from google authenticator from mobile. This TOTP will be change in every 30 seconds as shown in Figure 34.

Figure 34: google authenticator

After getting 6 digit time based one time password (TOTP), we will put it on verification code as shown in Figure 35.

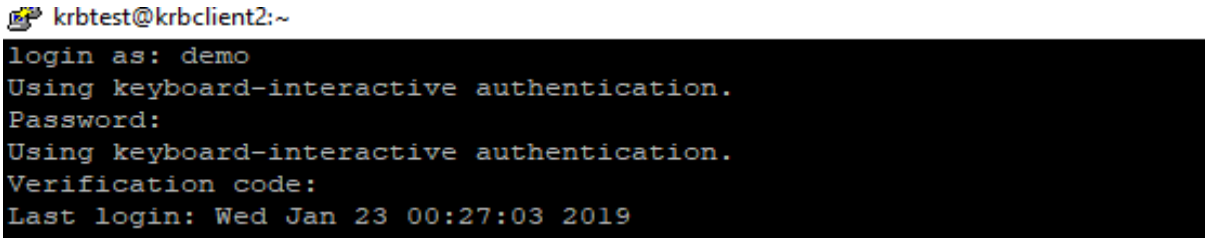

Figure 35: Verification code

Once matched the TOTP with server, it will allow successful login. Now we will 'su' to krbtest user. This users has created in Kerberos server. This user will prompt for password .Once password is authenticate from Kerberos server, it will be authenticated as shown in Figure 36. After that, we will run "klist" for getting ticket as shown in Figure 36. This ticket will be live for 1 minute. We will "ssh" Kerberos client2 "krbclient2" by using "ssh krbclient2.multias.com" and it will login to krbclient2 in single sign on as shown in Figure 37.

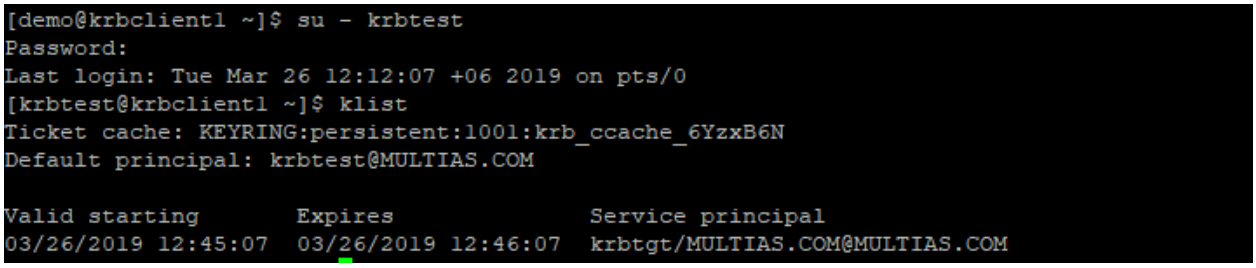

Figure 36: Kerberos user login and ticket generating

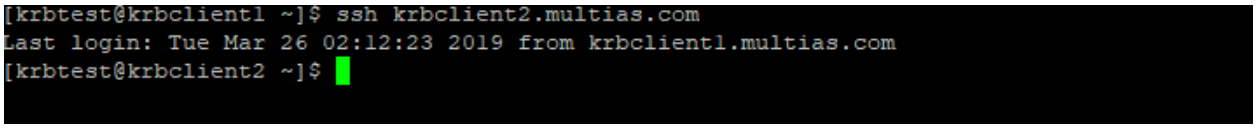

Figure 37: login krbclient2 by single sign on

We have successfully tested 3 level authentication password, verification code and single sign on.

Chapter 6

Results and Discussion

Three (3) levels of authentication are found in Figure 32, 33 and 37, respectively. These are password, Verification code and single sign on.

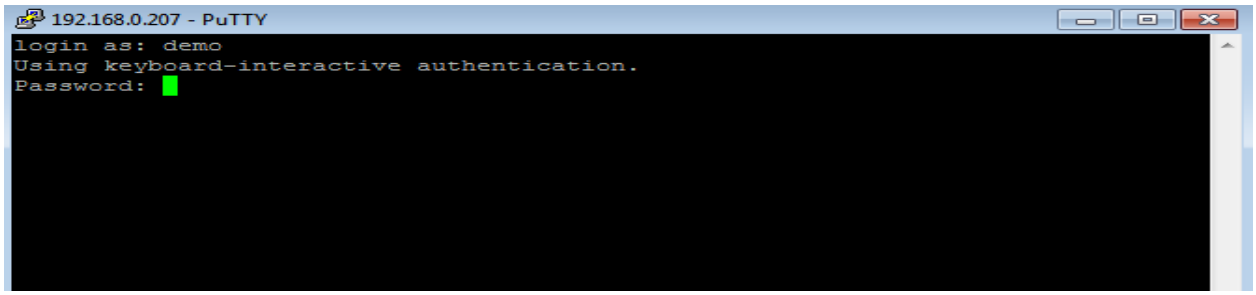

Figure 32: password

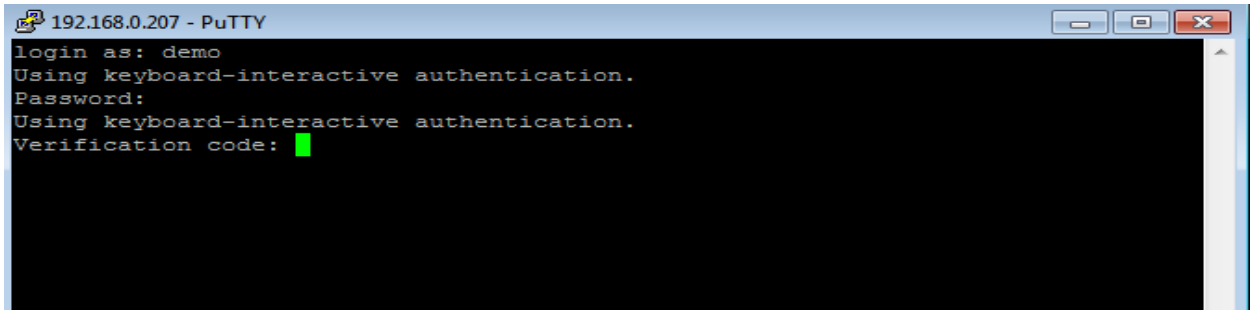

Figure 33: prompt for verification code

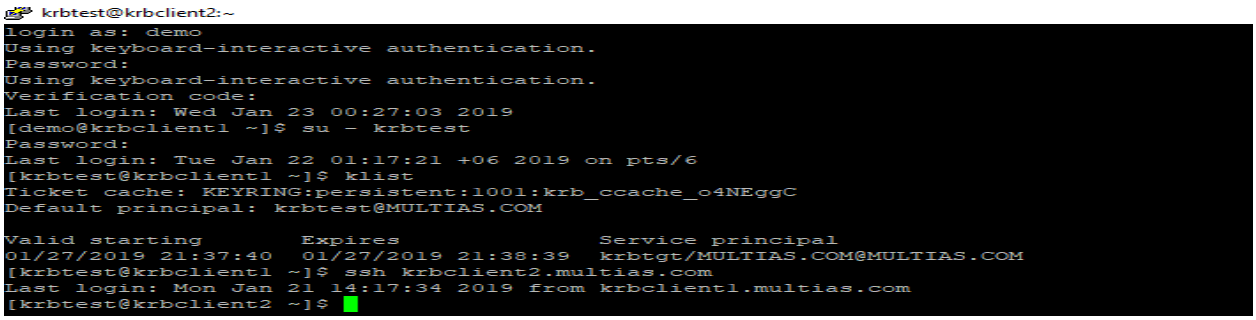

Figure 37: login krbclient2 by single sign on

By using three level authentication UNIX admin can do their data center administration securely. In this way, they will be secured from password theft, man in the middle attack, snooping and any kinds of unauthenticated situation.

In this project we have developed multilevel authentication for data center administration. Since data center is heart for any organization because server, storage run from data center. Therefore we have to operate data center securely. For this reason multilevel authentication is highly required for data center administration. In multilevel authentication we have used password, verification code, single sign on and syslog server to collect logs. We have used secure shell (ssh). Secure shell (ssh) is cryptographic network protocol for operating network service over network. We have used google authenticator for verification code. The google authenticator runs on server side and generate token with specific period of time and google authenticator runs on mobile that generate time based one time password(TOTP) in every 30 seconds. The server side and mobile time are synchronized with same time and no network connection required. Since there is no network connection required, it is secured from man in the middle attack, snooping. We have used Kerberos system for single sign on. Kerberos prove identity during node communication over network.

Our developed multilevel authentication system will provide high security than tradition login system for UNIX administrator for data center administration purpose.

Chapter 7

Conclusions

The multilevel authentication system provide strong protection since administrator will be prompted for password, verification code and single sign on. Once three level will be matched with authenticate server, administrator will be allowed to access server for administration in data center. In multilevel authentication system we have used password, TOTP, single sign on. The data center is center point for banking or organization to store data and access data from data center. A business relies heavily upon the application, services and data contained with data center, making it focal point and critical assert for everyday operation. Therefore it is very crucial to do the activities in data center securely. So, three level authentication password, verification code and single sign on will provide high security for UNIX admin to do their data center administration. In future we will enhance the security level more for this authentication system

References

[1] Technopedia, (January 21, 2019) Authentication, What does authentication mean [Online]. Available:https//www.techopedia.com/definition/342/authentication

[2] TOTP [Online]. Available: https://searchsecurity.techtarget.com/definition/time-based-one-time-password-TOTP

[3] Jiliang Zhang, Xiao Tan, Xiangqi Wang, Aibin Yan, Zheng Qin "T2FA: Transparent Two-Factor Authentication" in conf. IEEE, 15 June 2018

[4] Jaesik Lee, Youngseok Oh "A Study on Providing the Reliable and Secure SMS Authentication Service" in conf. Intl Conf on Ubiquitous Intelligence and Computing, IEEE, 9-12 Dec. 2014, Bali, Indonesia

[5] Wikipedia, Man-in-the-middle attack, [Online]. Available: https:// en.wikipedia.org/wiki/Man-in-the-middle_attack.

[6] Scott Ruoti, Jeff Andersen, Kent Seamons "Strengthening Password-based Authentication" in conf. Twelfth Symposium on Usable Privacy and Security , United States,2016.

[7] Himika Parmar, Nancy Nainan, Sumaiya Thaseen "Generation of Secure One-Time Password Based on Image Authentication" in conf.The Fourth International Workshop on Computer Networks & Communications, October 2012.

[8] Fadi Aloul, Syed Zahidi , Wassim El-Hajj "Two factor authentication using mobile phones" in conf. International Conference on Computer Systems and Applications, IEEE, 10-13 May 2009, Rabat, Morocco.

[9] Eko Sediyono, Kartika Imam Santoso, Suhartono "Secure Login by Using One-time Password Authentication Based on MD5 Hash Encrypted SMS" in conf.

[10] Marshall Brain, Stephanie Crawford, How Domain Name Servers Work [Online] Available: https://computer.howstuffworks.com/dns.htm.

[11] Wikipedia [Online] Available: https://en.wikipedia.org/wiki/Kerberos_(protocol).

[12] P. Iyappan,K. S. Arvind,N. Geetha,S. Vanitha "Pluggable Encryption Algorithm In secure Shell(SSH) Protocol" in conf. 2009 Second International Conference on Emerging Trends in Engineering & Technology Nagpur, India 16-18 Dec.2009 IEEE 10.1109/ICETET.2009.180.

[13] Tila Lodha 17th april 2018 Google Authenticator and how it works? [Online]. Available:https://medium.com/@tilaklodha/google-authenticator-and-how-it-works-2933a4ece8c2.

[14] Technopedia Pluggable Authentication Module [Online].Available: https://www.techopedia.com/definition/26088/pluggable-authentication-module

[15] Wikipedia syslog [Online] Available: https://en.wikipedia.org/wiki/Syslog

[16]MIT Kerberos Consortium, Ver. July23, 2008 [Online] Available: https://www.kerberos.org/software/adminkerberos.pdf.

[17] MIT Kerberos [Online] Available: https://web.mit.edu/kerberos/krb5-1.5/krb5-1.5.4/doc/krb5-install/What-is-Kerberos-and-How-Does-it-Work_003f.html.

[18] Mohamed Hamdy Eldefrawy, Khaled Alghathbar, Muhammad Khurram Khan "OTP-Based Two-Factor Authentication Using Mobile Phones" in conf. Eighth International Conference on Information Technology: New Generations, IEEE, 11-13 April 2011, Las Vegas, NV, USA.