Trinity College Dublin

Coláiste na Tríonóide, Baile Átha Cliath

The University of Dublin

Security and Privacy

ER -2

Declaration: "I have read, and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at http://www.tcd.ie/calendar. I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write,' located at http://tcdie.libguides.com/plagiarism/ready-steady-write."

Submitted By

Harsh Dhingra

19323904

Q 2 (a)

The Internet Key Exchange (IKE) is designed to create both inbound and outbound security associations. Kerberos a computer network authentication protocol that works based on "tickets" to allow nodes to communicate over a nonsecure network to prove their identity to one another in a secure manner.[1] It's a client-server network authenticator. Kerberos requires a trusted third party, which is the Key Distribution Center (KDC) for Kerberos. KDC has two different techniques; one is an authentication server commonly called AS and a Ticket granting server (TGS). KDC contains a database of secret keys. KDC generates a session key for the communication process that communication parties use to encrypt their transmission (now, since the transmission is encrypted, it would be safe in a nonsecure channel). The KDC protocol also makes sure that it's not transferring unencrypted keys over the network. A Kerberos realm is a place where all authentication takes place. It's a group of systems over which KDC can authenticate users and services.

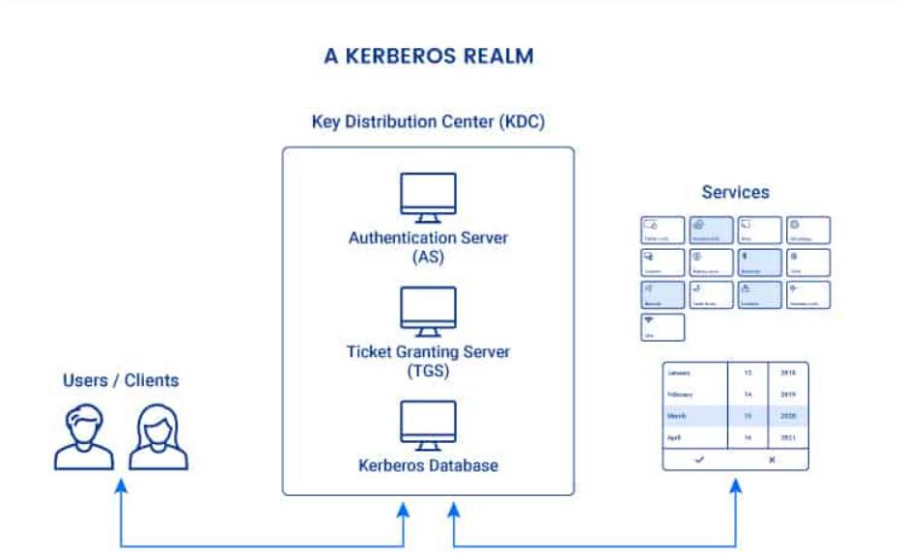

Fig (a) https://phoenixnap.com/blog/kerberos-authentication

Kerberos helps in eliminating the most challenging aspect of cryptography, i.e. Key management. Kerberos helps manage key as a trusted entity by users and server (or service. Therefore KDC (Key Distribution Center) is protected from any unauthorized access because KDC manages a database of keys

In Kerberos, AS (Authentication Server) Stores shared secret keys. Clients authenticate to AS and receive the long-term key. Now, whenever a client needs a service that wants to connect with the server, a request is sent to AS- it generates a session key(random) which is encrypted with the service name, a long term key provided by a client called credentials, and session key and principle name encrypted with service long term key called ticket. The client sends a timestamp (encrypted with session key) along with the Ticket to the service. Now the service decrypts the tickets, and after authentication, the connection is made. [8].

In Kerberos, there are three sub-protocols, namely the authentication service exchange (AS), the Ticket granting service exchange (TGS), and the client-server (AP) exchange, and each contains pair of message explained in Fig(B)

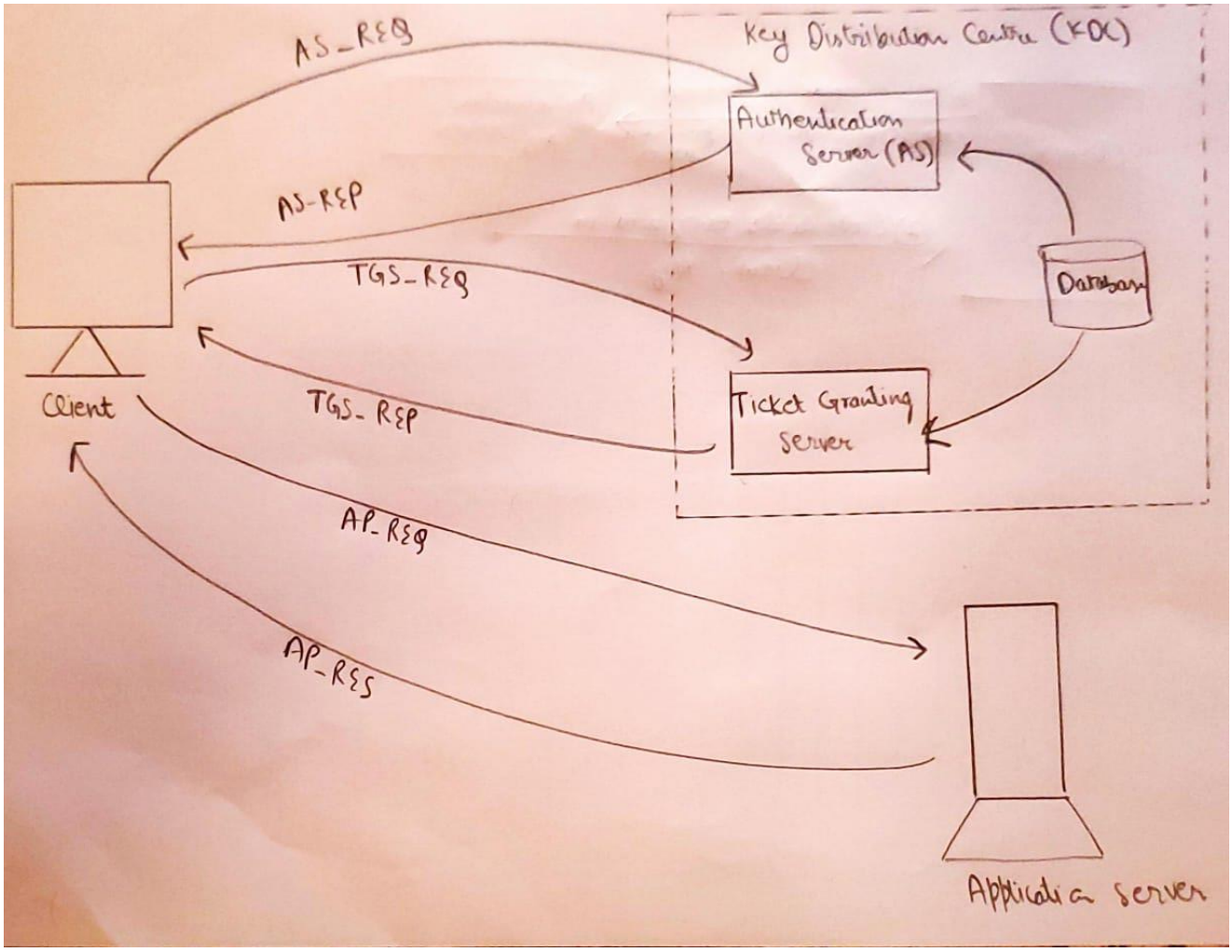

Fig (b)

- Step 1: Client sends a request to Authentication Server (AS) requesting access/credentials for a given server **AS_REQ** It requests a ticket from AS for TGS.
- Step 2: AS response to the client with credentials encrypted in the client's key. **AS_REP.** Receives a TGT encrypted with the user session key
- Step 3: Client sends the Ticket Granting Server – the Ticket granting Ticket, requests authentication. **TGS_REQ**
- Step 4: Ticket Granting Server to the client sends new session key and Ticket for application server **TGS_REP**

After these exchanges, the client authenticates its identity **AP_REQ** and can access the server **AP_RES**.

Imagine a user and user wants to gain access to network services (the user can be anywhere). It would be using the internet, which is a nonsecure channel, so to make the service with the network encrypted, we use the concept Kerberos. Before the user gets access to the network services, the user needs to have a ticket granted from KDC (Key Distribution Center), and the Ticket should be presented in the network (i.e., authentication via 3rd party KDC in this case) once its authenticated the network is provided to the user.

Misconfigurations in case of Kerberos Version 5:

1. **ENCRYPTION TYPE MISMATCHES**

   In Kerberos 5, there is extensible encryption support. It might result in some errors when mixed with different Kerberos 5 implementations; most of the time, KDC can automatically determine the optimal encryption type for a protocol exchange but can sometimes be seen. Usually, when KDC returns a ticket to the client for use with the server, it uses the most robust encryption type present in the database. Due to a case of mismatch, e.g., if keytab(in-service) entry contains a single DES key and KDC has issued the client the service ticket with triple-DES – in this case, the service can't find the appropriate encryption key to decrypt the Ticket; therefore, it returns an error message.
   The solution would be to identify the encryption type supported by the service and ensure that the Kerberos database contains only those encryption types.

2. **UNSYNCHRONIZED CLOCKS**

   Another common problem in the case of Kerberos is unsynchronized clocks, and it occurs due to the lack of all participating hosts
   The error message produced is :

```
krb5_rd_req failed: Clock skew too great
```

   Therefore it's recommended that all the participating hosts in a Kerberos realm be synchronized to a central time source

Q 2 (b)

MIT Kerberos

MIT Kerberos was the first and is a reference implementation of both Kerberos Version 4 and Kerberos Version 5. It contains support for standard Kerberos encryption types , like single and triple DES. It uses strong cryptography so that the client can prove its identity to a server. MIT Kerberos is freely available under copyright permissions MIT ‘s new release is Kerberos 5 1.19.1 which uses GSSAPI krb5 mech

GSSAPI – Generic Security Services Application Program Interface library in MIT Kerberos5 can load mechanism modules to augment set of mechanisms. GSSAPI does not provide securitu itself but it’s a framework that provides security services to the callers – it’s a generic authentication and secure messaging interface It enhances the existing key management.
Advantages to use GSSAPI Kerberos –

- Good cross platform availability.
- Good interoperability.
- Support for future extensibilty.

Q 2(c)

Kerberos as its build on symmetric key infrastructure and cross realm it is considered safe from quantum attacks as long as we avoid public key. One solution could be we should not rely on public key crptography and instead of it we should mix secret entropy that couldn’t be decipher in case of quantum attack We can use Hash- Based cryptography it’s a quantum proof cryptographic scheme – so the keys used in authentication should be encrypted using this technique.

References

[1]https://sc1.checkpoint.com/documents/R80.10/WebAdminGuides/EN/CP_R80.10_IdentityAwareness_AdminGuide/html_frameset.htm?topic=documents/R80.10/WebAdminGuides/EN/CP_R80.10_IdentityAwareness_AdminGuide/101050

[2]https://www.accenture.com/_acnmedia/PDF-87/Accenture-809668-Quantum-Cryptography-Whitepaper-v05.pdf

[3] https://learning.oreilly.com/library/view/kerberos-the-definitive/0596004036/

[4]https://datatracker.ietf.org/doc/html/rfc2743.html#page-89

[5]https://www.informit.com/articles/article.aspx?p=102612&seqNum=3

[6] https://web.mit.edu/kerberos/

[8] https://waynefischer.weebly.com/key-management-kerberos.html