International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 02 | Feb 2022 www.irjet.net p-ISSN: 2395-0072

Kerberos Security in Distributed Systems

Pratik Joshi

[1]*Pratik Joshi; Stevens Institute of Technology, New Jersey, USA*

Abstract - *Most institutes use web applications to get access to their clients' real-time information. Given the amount of data we have, we need multiple systems to process and hold this data. In a distributed system, the work is split into multiple computers making it scalable in near real time, helping in increasing the performance and reducing time to completion.*

They use this data to carry out various transactions and perform several key operations. However, if that application does not have proper security features and was not created with efficient coding, this may lead to cyber-attacks. Kerberos is a network authentication protocol that centralizes the authentication database and uses applications to work with servers or services. This way it supports allowing single logins and encrypted communication over internal networks of the company. It provides the means to authenticate clients to servers, and in a secure way. It also provides support to other services that have proven both useful and vital to achieving security objectives within today's information systems.

Key Words: Kerberos, Key Distribution Center (KDC), realm, authenticator, encryption, shared secret key.

1.INTRODUCTION

Achieving security in today digital age has proven to be a very hard problem. In many situations, security measures are applied as an afterthought once the incident has been reported. Instead of taking a comprehensive measure while developing the system our field has deployed security measures and technologies at need basis. Over time, this patches for security in application system resulted into mosaic of security-related technologies making it brittle, difficult to extend, time-consuming to manage, and redundant.

In this document we will learn about Kerberos as a trusted third party service and how we can use that in our modern day distributed system. We will take an architectural look at security, discuss process of integration and best practices. It focuses on the role of the Kerberos authentication system as part of the overall ecosystem likely to be encountered in a typical modern distributed-systems environment. This is intended for engineers who like to design, manage, and build applications taking security into consideration.

1.1 Introduction to Kerberos

The Kerberos Authentication System uses a bunch of encrypted messages. These messages are used to prove to a verifier that a client is running for a user. The Kerberos protocol is designed to provide secured authenticated communication over open and insecure networks. Kerberos support the needs of the environment for which it was developed and make changes like using the timestamps to reduce the number of messages needed for authentication. A realm is an authentication administrative domain who establishes boundaries within which authentication server has authority to verify identity of the user. The addition of a "ticket-granting" service helped in achieving authentication between users and services across realms, this is known as cross-authentication. A user or service is part of a realm if they share key with authentication server of that realm.

A client sends an authentication request to Authentication Server (AS) which is part of the KDC (Key Distribution Center). In reply to this AS sends service ticker and session key to the client. Using this service ticket, the client can now send a request to service that he is trying to use.

One of the key features of DES is that if cipher text is decrypted with the same key used to encrypt it; you see the original data. Kerberos uses this implementation called DES (Data Encryption Standard) to encrypt data. If different keys are used for encryption and decryption, or if the text is modified, the result will not match the data. This combination of encryption and the checksum on data provides integrity and confidentiality for encrypted Kerberos messages. The client and server do not share an encryption key at first. Whenever a client authenticates to a new Authentication Service it generates a new encryption key and distribute it securely to both parties. This new key is called a session key and they do not change even when work session changes. They are also known as long term.

1.2 History of Kerberos

Developed in MIT, Kerberos is meant to protect network services. In Kerberos Version 5, the intention was to overcome the limitations and security problems of version 4. MIT made Kerberos freely available, under copyright permissions and formed the Kerberos Consortium to promote development. Founding sponsors include tech giants like Sun Microsystems, Apple, Google, Microsoft and Centrify Corporation, and academic institutions such as Stanford University and MIT. Some of the key features introduced in V5 are cross-realm authentication, facilitates forwarding tickets.

2. How does Kerberos V5 work?

To access network services Kerberos issues tickets for authentication. These tickets hold encrypted data, including

an encrypted password that verifies the user's identity to the service being requested. The entire authentication process is invisible to the user except for the part where the user enters a password.

A key service within Kerberos is the Key Distribution Center (KDC). The KDC is running on each distributed network as part of the Active Directory service. This AD stores all passwords and other account information.

The Kerberos authentication process works as follows:

1. The user on a client system, authenticates to the KDC using the password as user key
2. The KDC takes in the details and issues a special ticket-granting ticket to the client. Using the TGT client access the ticket granting service (TGS), which is part of the Kerberos V5 authentication mechanism on the domain controller.
3. The TGS then sends a service ticket to the client.
4. The client presents this service ticket that he received from TGS to the requested network service. The service ticket authenticates both the user's identity to the service and the service's identity to the user.

The way Kerberos V5 services work is by installing the service on each domain controller, and a client is installed on each workstation and server. Every domain controller acts as a KDC. A client uses a Domain Name Service (DNS) lookup to locate the nearest available domain controller. That domain controller then processes as requested by the preferred KDC for that user during the user's logon session. If the preferred KDC becomes unavailable, the system locates an alternate KDC to provide authentication.

2.1 Why do we need keys?

The Kerberos protocol uses a shared secret key for its authentication. Shared secret key works on basic principle that only two parties involved in the communication would know the secret and can verify the identity before initiating the communication. The Kerberos protocol makes it better by using a secret key cryptography. Instead of sharing a password, share a cryptographic key, and they use that key to verify the identity before starting the communication. For the technique to work we still need the shared key to be symmetric—that is, a single key capable of both encryption and decryption. One entity proves they have secret key by encrypting some information and the other entity proves knowledge of the same key by decrypting the information. If the same key is used the encryption and decryption would work and verify that it is a secret between the two entities. Kerberos authentication relies on bunch of keys and various key types for encryption. There are key types like long-term symmetric keys and asymmetric keys, and short-term symmetric keys like a secret key.

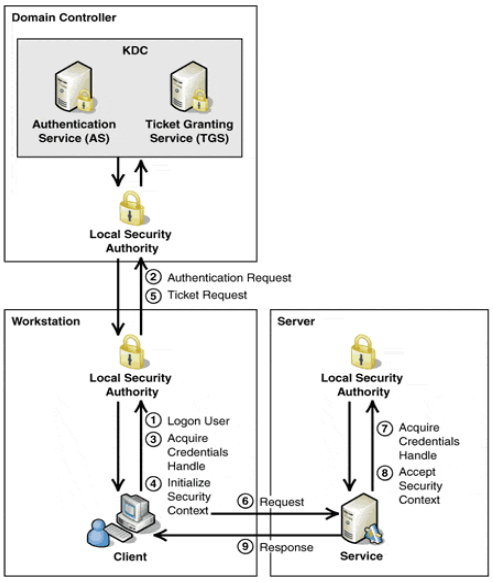

Fig 1: Working of Kerberos V5

User keys: When a user is created, user key is created from the password. In the authentication service database like active records, the user key is stored with the user's object. At logon to the workstation the user key is created.

System keys: When a server joins a Windows domain, it gets a password. In the same way as a user account, the system account's password is used to generate the system key.

Service keys: Based on the account password that is used to log on, services use a key. All Key Distribution Center's in the same realm use the same service key.

Inter-realm keys: For cross-realm authentication to take place, the KDCs must share an inter-realm key. The realms can now begin to communicate and trust each other because they share the key.

2.2 Ticket-Granting Tickets

In response to client's authentication service the KDC responds by returning a service ticket for itself. This service ticket is called a ticket-granting ticket (TGT). A TGT enables the authentication service to securely send the requester's credentials to the ticket-granting service.

A TGT is like a first step in the authentication service, it facilitates service tickets to the client and generates tickets that can be used by ticket-granting service. TGTs are usually encrypted with a key shared by the KDCs. The client is unable to read those tickets and only KDC servers have capability to read TGTs to secure access to user credentials, session keys. Like any other service ticket, a TGT holds a copy of the session key that the service (in this case the KDC)

will make use of for communicating with the client. The TGT is encrypted using the long-term key obtained from the KDC.

The Authenticator

Whenever a client sends a ticket to a target server it includes an authenticator in the message. That authenticator is the means of identification, and it can originate from the server inside the TGS or some other network server or resource. The authenticator verifies that the client who is listed as the sender in the ticket belongs to the ticket's source.

Why is an authenticator necessary?

The target server's secret key is used to encrypt the contents of the ticket because the ticket is encrypted. However, the target server cannot be sure that the ticket was really sent by the client mentioned in the ticket. There could be an attacker impersonating the client and had access to the ticket.

How does the authenticator work?

- Using the session key generated by KDC the authenticator is encrypted with the session key created by the KDC to be used between the client and the target server. The client and the target server are the only ones who can have access to this session key.

- The secret key is used by the target server to decrypt the ticket. The target server finds the session key inside the ticket and uses it to decrypt the authenticator.

Once the target server can successfully decrypt the authenticator and if the data is accurate, then the target server will process and trust the source of the ticket.

2.3 Kerberos V5 protocol standards

To verify the identities of the users on an open network, Kerberos need to provide a mechanism. It achieves this without relying on users, or his workstation or infrastructure he is part of. It does assume that packets transferred over the user's network can be intercepted, read, modified by anyone. Kerberos uses shared secret key cryptography to perform authentication. There are implementations where public key cryptography is supported for users registered with public key.

2.4 How to integrate Kerberos into your application?

Due to its ticketing system, Kerberos does not need pass-through authentication which helps accelerate the authentication process. Let's take a closer look at how Kerberos work in detail.

Setting up the authentication server

A database of all the users and their password is stored at a site on a machine which is extremely secure, important, and well protected. This machine should be dedicated to run services to authenticate users who try to authenticate themselves. A default password is sent to all the users and on first login, users can set their own password, and these are registered with authentication server. If the number of users using are limited, we can physically verify each of the users and provide passwords. But for big firms who had thousands of users, we would need a team for administrative work.

Application designers need to look how will Kerberos security interact with the rest of the application, how will the application use Kerberos, when to make a call to authentication server, and which route should be taken to integrate Kerberos? They need to know what Kerberos assumes about the client that ties to connect. Make appropriate provisions if these assumptions are not correct and disconnect. Let's take a closer look on some of the best practices to keep in mind on how we can use Kerberos and integrate into our application. The application itself should be secured and have all security measure in place before it initiates the communication with Kerberos system. The network in which these services are running should be secured. Kerberos should handle proper authentication and session security services. Kerberos authentication should not rely on other services to authenticate users. When an application supports authentication alongside Kerberos, a failure in the Kerberos should not lead the applications to fail for users who do not use Kerberos. There should be clear boundaries to avoid failures for non-Kerberos user. The application should take full advantage of security provided by Kerberos and not try to work around or override their implementations. There shouldn't be any direct interaction of user with Kerberos, it is best practice to have a service communicate with Kerberos service. Appropriate integrity and confidentiality services are provided depending on the access control policy of the data being transported over the network.

Various applications already have adopted various strategies for integrating Kerberos that meet most of these practices. If Kerberos has already been integrated into applications, try to follow these best practices. For protocols like SMTP, LDAP, or other protocols that are implemented by application services, adopting a common approach is essential. This is to make sure that applications from one vendor can work with applications from another.

2.5 Limitations of Kerberos

A major drawback of Kerberos ticketing system is that it increases computation load on the client. It can be a standalone service for authentication but that comes at a price of latency. Though this latency is not much, it could become problem for trading applications where volume and efficiency are required.

Kerberos is not effective against password guessing attacks; an attacker guessing that password can impersonate the user. Similarly, Kerberos requires a secured network through which passwords are entered. If the user enters a password to a program that has already been compromised, then an attacker may obtain sufficient information to impersonate the user. To overcome this brute force attack Kerberos must be integrated into application with additional security measure. It should take care of initial authentication and then rely on tickets for any ongoing communication. While it may be used to exchange encryption keys on establishing initial encryption and network level security services, this would require changes to the network software of the user.

The KDC should be always available in Kerberos, which makes the design susceptible to single point failure. Whole authentication service could come to a stand still if KDC is unavailable.

Kerberos only supports client to server communication but in a distributed environment we would like services to talk to each other and not just the client.

2.6 Applications of Kerberos

Since Kerberos Authentication Server is an entry point for all authentication request, it can provide a good access control system to admins. It can be used to keep a track of logins and security enforcement acting as an effective Access Control system.

Using Kerberos authentication within a domain allows the user or the service that tries to access the resources permitted by administrators without multiple requests for credentials. After initial domain sign on through Win logon, Kerberos manages the credentials throughout which forms the basis of Single sign-on (SSO).

Mutual authentication can be achieved by verifying the other party since we share the same secret key. This can be used to ensure identity of the services within a network.

Delegating authentication for authentication can make Kerberos authentication useful in distributed system. Say a browser, webserver, and database server are all running on different systems, authentication can occur at different tiers. User identity can be verified by querying database or exposing service to call the database thereby creating a boundary between systems.

3. CONCLUSIONS

Kerberos with use of right design can overcome many shortcomings we discussed before. Using machine learning we can make our KDC more robust and help analyze any anomaly in the behavior. This can help in flagging unusual activity from intruder, since we have more data points to understand each request. Using rate limit technique on KDC can help overcome the password brute force attack. Even if the user has a weak password, we can limit the number of request each client can make within a given timeframe. To overcome spoofing while login, a onetime session approach can be used. A user and server can share a secret key through another medium like a device or file.

We reviewed Kerberos as an authentication service, to realize the working with the help of tickets and keys and the relevant security identification, starting on prevention from the external factors of system security, through adequate improvement on protocol architecture and process to effectively remedy the existing defects in the protocol itself, and thus enhancing the security of the financial system. Many institutions have opted for two-factor authentication (2FA) as a measure to increase security. After reviewing Kerberos design, we now have a better understanding of how it works, the benefits of the service and how it can be integrated into our application.

REFERENCES

[1] Clifford Neumann. The Kerberos Network Authentication Service (V5). Internet Draft ietf-cat-kerb-kerberos-revision-04.txt, June 1999

[2] Kerberos: A review of the modification in Versions 4-To-5 Transition – B.A. Andrew

[3] http://www.kerberos.org

[4] RFC1964: The Kerberos Version 5 GSS-API Mechanism

[5] RFC4120: The Kerberos Network Authentication Service (V5) [Applied Cryptography] Second Edition, Bruce Schneider

[6] Tom Yu, Sam Hartman, and Ken Raeburn. The Perils of Unauthenticated Encryption: Kerberos Version 4. In Proceedings of the Network and Distributed System Security Symposium. The Internet Society, February 2004.

BIOGRAPHIES

I have been in Finance and Technology for over 9 years. At MarketAxess, I design solutions for our leading electronic trading platform for fixed-income securities. My team and I manage the market data and post-trade services for the global fixed-income markets. I work on making the trading platform that sees on an average $300 billion monthly volume efficient and optimized.