International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

A Survey on Authenticated Key Exchange Protocols Used for Accessing Storage Devices in Parallel Network File System

Anila P[1], Prajeesh C B[2]

[1]MEA College of Engineering, Kerala Technological University, Vengoor, Perinthalmanna, Malappuram, Kerala, India
[2]MEA College of Engineering, Calicut University, Vengoor, Perinthalmanna, Malappuram, Kerala, India

Abstract: ***Key generation for secure many to many communications is a major problem addressed in accessing storage devices in parallel Network File System (pNFS). It use Kerberos to establish parallel session keys between client and storage devices By reviewing Kerberos based protocol it has been find that there are a number of limitations such as unscalable, lack of forward secrecy and key escrow. In this work I propose an authenticated key exchange protocol that is designed to address the above limitations of Kerberos. It can be show that this protocol will reduce the workload of server thereby achieving forward secrecy and the key become escrow free.***

Keywords: NFS, pNFS, Kerberos, LIPKEY, key escrow, forward secrecy

1. Introduction

File system is a method uses to keep track of files on a disk or partition. The Network File System (NFS)[2] is a file system developed by Sun Micro Systems. It is used to provide distributed transparent file access in different types of network. NFS is an industry standard which is able to share entire file systems among machines within a computer network.

For using NFS on the internet, it has been developed the new version NFS version 4 called Parallel Network File System (pNFS) which is developed to;

1. Improve the accessibility and performance
2. Improve the security
3. Improve interoperability

In the parallel file systems the files are distributed among multiple storage devices or nodes that can allow the concurrent access of multiple tasks of a parallel program. This is mainly used in large-scale cluster computing that needs high performance and reliable access to large datasets. That is, it can achieve higher I/O bandwidth through concurrent access to multiple storage devices within large clusters. [1]

In this work, I am trying to find out the problem in the security of communication in parallel NFS. That is, I consider a model that consists of a number of users trying to access multiple storage devices in parallel. The main focus is on the method of exchanging the keys and establishment of parallel secure sessions between the user and the storage devices. The primary goal is to study different authenticated key exchange protocols and find out which method is more secure than the others.

2. NFS

The Network File System (NFS) [1] protocol is a distributed file system protocol that was developed by Sun Microsystems. It allows a user on a client computer, which may be diskless, can access files over networks in a manner similar to how local storage is accessed. One major advantage of NFS is the capability of central management. The centrally managed server decreases the workload for the administrator since the server will take care of back-ups, adding software that need to be shared. Another advantage is that even a small portion of file can be accessed by the authorized user. NFS allows the user to log into the server and can access the files transparently. The main disadvantage of NFS is the problem of security. Since NFS is based on remote procedure calls, it is probably insecure and it can be only be used on the trusted network behind a firewall. The other disadvantage is the performance limitations on the network. NFS will slow down during the increment of network traffic.

3. pNFS

The most recent version NFSv4.1[3] (NFS version 4.1) protocol provides a feature called parallel NFS (pNFS) that allows direct and concurrent client access to multiple storage devices to improve performance and scalability.

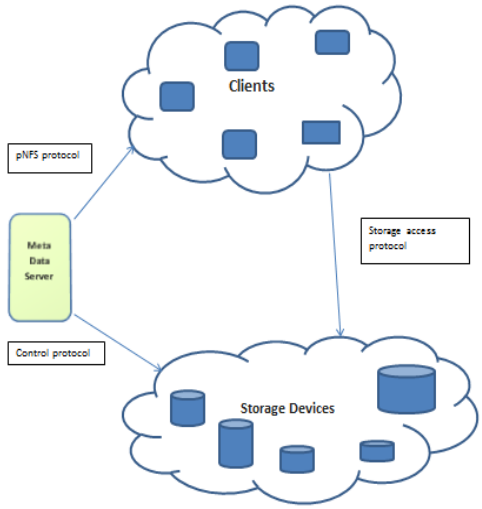


Figure 3.1: Parallel Network File System

The figure 3.1 shows the architecture of pNFS. One benefit of parallel NFS is that, the application server or client can gain simultaneous access in parallel over multiple data path to storage servers or nodes. A Metadata server will give location of the data to the client. The client can read and write the data directly to the storage.

Parallel NFS is important because it brings together the benefits of parallel I/O with the benefits of the ubiquitous standard for network file systems (NFS). Thus the pNFS will increase the performance and scalability in their storage devices and give guarantee that the added content is secure.

The current design of NFS/pNFS [2] mainly focuses on interoperability than that of efficiency and scalability. Moreover, the establishment of key between a client and multiple storage devices in pNFS are similar to NFS. That is, they are not designed any specific key establishment for parallel communications. Hence, the metadata server is mainly responsible for two things:

- Handling the request that send by the client for the accessing the storage devices, by giving valid layouts to authenticated and authorized clients
- Generating all the corresponding session keys that the client needs to communicate securely with the storage devices to which it has been permitted access

Therefore, the metadata server may become a performance bottleneck for the parallel Network File System. This protocol design may leads to the problem of key escrow. Moreover, by using this protocol the server will contain all the information transmitted between a client and a storage device. Therefore, the attackers will make an insight to the server so that they can hack every information from the server itself.

Another drawback of the current approach is that past session keys can be revealed if a long-term key of a storage device that shared with the metadata server is compromised. This is a major problem since pNFS may have thousands of geographically distributed storage devices. It may not be possible to provide strong physical security and network protection for all the storage devices since there are huge numbers of storage devices.

Key-exchange protocols are mechanism of creating a common secret key for the communication of two parties through a controlled network. Key-exchange protocols are important because it is used for enabling shared-key cryptography. The shared key cryptography protects the transmitted data from unauthorized access over insecure networks.

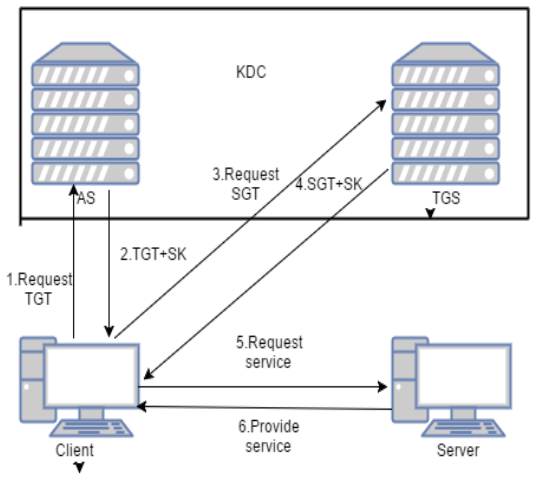
4. Literature Survey

4.1. Kerberos

Kerberos [4] is a computer network authentication protocol that helps people from purloin information's that sent across one place to another place in computer networks. In Kerberos based solutions, the communication between client and storage device is taken place through the metadata server which is capable of generating the authentication key. This Kerberos works on the basis of tickets that prove the identity of people who communicate over a non-secure network in a secure manner. Kerberos is a centralized network authentication system developed by MIT. It is the default authentication method in Windows 2000 and later. Kerberos uses symmetric key cryptography and it requires a trusted third party. Kerberos can also use public-key cryptography during certain phases of authentication if needed.

Scalability, Forward secrecy and Escrow free are some of the desirable properties which are not achievable by this Kerberos-based solution and also they are weak in provide security to the long-term keys. Forward secrecy means provide security to the past session keys when long-term keys are used. In key escrow the keys which are needed to decrypt or encrypt data are held in escrow so that under certain circumstances an authorized third party may gain access to those keys. It should be protected.

In Kerberos protocol, the client will repeatedly authenticate to multiple servers by assuming that there is a long-term secret key shared between the client and Kerberos infrastructure. The long-term secret key of client was generated by using the client's password. A simplified overview of the Kerberos actions is shown in Figure 4.1. The communication between the client and the Kerberos Authentication Server (AS) in Messages 1 and 2 are occurs only when the user first logs in to the system. Here, the client will request for the Ticket Granting Ticket (TGT) to the Authentication Server (AS). The AS will respond to the client by sending the TGT and the Session Key (SK). After getting the TGT and SK the client will request for a Server Granting Ticket (SGT). The Ticket Granting Server (TGS) will respond to the client through SGT and SK. This communication between the client and the Kerberos Ticket Granting Server (TGS) in Messages 3 and 4 are used whenever a user is finding to be a trusted person to a new server. Message 5 is used to request for a service. This will occur whenever the user authenticates itself to a server and Finally, Message 6 is the message sent by the server, the authentication response to the client as the requested service.



AS →Authentication Server
TGT → Ticket Granting Ticket
KDC →Key Distribution Center
TGS →Ticket Granting Server
SGT →Server Granting Ticket
SK →Session Key

Figure 4.1 Overview of Kerberos authentications.

4.2 Limitations of Kerberos

1. Kerberos [5] is sensitive to password guessing attacks. The Kerberos will have an encrypted message with a key that uses the client's password for encryption. The attacker can hack this message and try to decrypt it by guessing different passwords. If the result of guessing is in proper form, then the attacker can discover the client's password and they may use this password to gain authentication credentials from Kerberos
2. The system clocks of the hosts should be synchronized [5]. Otherwise, the time in the host does not match with the availability period the Kerberos server clock, and the authentication will get fail.
3. KDC should be always available [5]. When the KDC is down, the entire system will suffer from the single point of failure problem.
4. Difficult to achieve Scalability[2] Since encryption of the message is done by the server itself and as the number of clients increase, the workload of server will increase and it limit the scalability of the system
5. Cannot achieve Forward secrecy [2] where Forward secrecy means provide security to the past session keys when long-term keys are used.
6. Cannot achieve Escrow freeness [2]. In key escrow the authorized third party can access the keys which are needed to decrypt or encrypt data and if the third party is not protected well the intruders can get the key and perform unauthorized access of storage devices in the case of Kerberos.

4.3 LIPKEY

SPKM (Simple Public Key Mechanism) is a GSS-API mechanism which is based on a public key mechanism [6]. SPKM provides authentication, establishment of key, integrity of data, and confidentiality of data using a public key infrastructure. SPKM data formats and procedures are designed as similar as Kerberos mechanism as is practical. SPKM is used in the applications that need to have a GSS-API mechanism based on a public key infrastructure.

LIPKEY (Low Infrastructure Public Key Mechanism using SPKM) GSS-API mechanisms, such as Kerberos Version 5 and SPKM requires infrastructure [6]. LIPKEY is a low infrastructure-based GSS-API security mechanism. It consists of a client who has no public key certificate and tries to access a server with a public key certificate. The LIPKEY mechanism can be used when the initiator (client) does not have a public key certificate. The user will have the username and password for authentication.

Working of LIPKEY:

- The client with no public key certificate sends the request to the server.
- The server who has the public key certificate passes its certificate as the reply token to the client. The client uses this certificate to verify the server.
- Then the client and server establish a secret session key using Diffie Hellman key establishment mechanism.
- The client then encrypts the user name and password and gives it to the server.
- The server decrypts the username and password of the client and validates that information to authenticate the client. The validation can be done by using the information in the centrally managed username and password database.

4.4 MAKEP (Modified Authentication Key Exchange Protocol)

This protocol [2] is a modified version of Kerberos that permits the client to generate the session keys instead of server thereby reducing the workload of server. It uses a Diffie Hellman key agreement technique to address the problem of key escrow. Here the client and the storage devices will have its own secret value and they pre-compute the Diffie Hellman key component. The session key is generated by using these two Diffie Hellman key components. Thereby reducing the key escrow since the server does not have any involvement in the session key generation. The protocol has a fixed time period .After that time period; client will not have access to the storage devices so that it can achieve full forward secrecy. That is, revealing the long term key affects only the current session key (with respect to that time period), but not the past session keys.

5. Conclusion

In this work I conclude that the MAKEP protocol address the limitations of Kerberos based protocol. This network and can protocol can increase the scalability of the network and can achieve the forward secrecy. The protocol can also provide the escrow freeness. The protocol is free from password guessing attack also. So that, it can be concluded that MAKEP is a better protocol than Kerberos.

References

[1] Pawlowski, Brian, et al. "The NFS version 4 protocol." Proceedings of the 2nd International System Administration and Networking Conference (SANE 2000). Vol. 2. No. 5. 2000
[2] Lim, Hoon Wei, and Guomin Yang. "Authenticated Key Exchange Protocols for Parallel Network File Systems." IEEE Transactions on Parallel and Distributed Systems 27.1 (2016): 92-105
[3] Kulkarni, Omkar, et al. "Study of Network File System (NFS) and Its Variations." Journal of Engineering Research and Applications (IJERA) ISSN2248 (2011): 9622

[4] El-Emam, Eman, et al. "An Authentication Protocol Based on Kerberos 5."IJ Network Security 12.3 (2011): 159-170.
[5] Bellovin, Steven M., and Michael Merritt. "Limitations of the Kerberos authentication system." ACM SIGCOMM Computer Communication Review20.5 (1990): 119-132
[6] Eisler, Mike. LIPKEY-a low infrastructure public key mechanism using SPKM. No. RFC 2847. 2000

Author Profile

Anila P received her B.Tech degree in computer science and engineering from the Mar Baselius College of Engineering And Technology and Technology, Kerala, in 2014. Right now she is pursuing her M. Tech degree in computer science at MEA Engineering College, Kerala from 2015 to 2017, Anila worked as a Program Analyst Trainee at Cognizant Technology Solutions (CTS) Tamilnadu. Her research interests lie in Network Security.

Prajeesh C B received his B.Tech. degree in computer science and engineering from the College of Engineering Adoor, Kerala, in 2007 and received his M. Tech degree in computer science at MEA Engineering College, Kerala in 2014 . Right now he is working as Assistant Professor in the Department of Computer Science at MEA Engineering College, Kerala. His research interests lie in Network Security and data mining.