Scientific Journal of Impact Factor (SJIF): 4.14
e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 3, Issue 3, March -2016

Secure File Sharing Using pNFS Protocol

Ghogare Jyoti[1], Mohite Komal[2], Salunkhe Komal[3], Zagade Sadhana[4] H. R. Kumbhar[5]

[1-4]*Student, Department of computer engg, SVPM's COE, Malegaon (bk),Baramati,Pune*
[5]*Professor, Department of computer engg, SVPM's COE, Malegaon (bk),Baramati,Pune*

Abstract—*The Large-scale distributed file systems supporting parallel access to multiple storage devices,in which the multiple client can access the server so there is problem of key establish-ment between client and server for secure communication. Our work focuses on the current Internet standard for such file sys-tems, i.e., parallel Network File System (pNFS), which makes use of Kerberos to establish parallel session keys between clients and storage devices. limitations of existing Kerberos based protocol shows that (i)The metadata server having heavy workload due to key exchange between the sever and clients,because of this scalabilty of of system get reduced.(ii)The kerberose protocol does not provide forward secrecy.(iii) The metadata server itself generates all the session keys used between the clients and storage devices,and this leads to problem of escrow free. In this system, authenticated key exchange protocols that are designed to overcome the limitation of the existing system.*

Index Terms—Scalability, Forward Secrecy, Escrow-Free, Au-thenticated key exchange.

I. INTRODUCTION

In parallel file system data or files store in multiple storage device or server. That allow the multiple node or client to access the multiple storage device.In parallel file system data is store in distributed manner.Distributed file system provides the high performance and reliable access of multiple storage device. pNFS is parallel network file system that allow the multiple client to share the data in network in secure man-ner.Some example of high performance parallel file system is Google File System,Parallel Virtual File System,IBM General Parallel File System.

Our primary goal is to design the efficient and secure authenticated Key Exchange Protocol that provides the spe-cific requirement of pNFS.The following desirable properties, which either have not been satisfactorily achieved or are not achievable by the current Kerberos based solution :

Scalability :The metadata server able to access request of clients for multiple storage device and it should not get workload,its able to support all request.the performance of server do not get bottleneck.
Forward secrecy :The protocol does not generate new session key from the previously generated session key or any information of session key.

Escrow-free :The metadata server should not be learn or known any information i.e session key between client and server communication.

In main result of this system are three available secure authenticated key exchange protocols. NFS is typically used in environment where the performance is major factor for ex. high performance, linux clusters.The NFS version 4.1 (NFSv4.1)[2] protocol, the most recent version, provides a feature called parallel NFS (pNFS) that allows direct, con-current client access to multiple storage devices to improve performance and scalability.

II. EXISTING SYSTEM

In kerberos protocol authentication is done on the basis of symmetric key rather than passwords.Client and server get communicate and share files using the cryptographic key. The symmetric key is used to identify one another's.the server can provide knowledge of key by encrypting pieces of information. the client by decrypting it.In kerberos KDC(key distribution center)is used that provide ticket grating service to client.The limitation of the kerberos are i)Scalability ii)Forward secrecy iii)Escrow free.

In kerberose the metadata server get overloaded,it does not able to handle request of no of clients simultaneously. Meta-data server generate the session key for the both clients and server which want to communicate and this leads to escrow free.[1]

@IJAERD-2016, All rights Reserved

A. Kerberos System:

In the existing system kerberos protocol is used.Kerberos is an authentication protocol which uses a shared secret and a trusted third party arbitrator in order to validate the identity of clients. In Kerberos, clients may be users, servers, or pieces of software. The trusted third party arbitrator is a server known as a Key Distribution Center (KDC) which runs the Kerberos daemons. The shared secret is the users password transformed into a cryptographic key. In the case of servers or software systems, a random key is generated. Kerberos is used particularly for user authentication and single sign-on.

User and Server Authentication:

Kerberos, a widely deployed network authentication protocol supported by all major operating systems, allows nodes communicating over a non-secure network to perform mutual authentication. It works in a client-server model, in which each domain (also known as realm) is governed by a Key Distribution Center (KDC), acting as a server that authenticates and provides ticket-granting services to its users (through their respective clients) within the domain. Each user shares a password with its KDC and a user is authenticated through a password-derived symmetric key known only between the user and the KDC. However, one security weakness of such an authentication method is that it may be susceptible to an off-line password guessing attack.

Single Sign-on:In NFS/pNFS that employs Kerberos, each storage device shares a (long-term) symmetric key with the metadata server (which acts as the KDC). Kerberos then allows the client to perform single sign-on, such that the client is authenticated once to the KDC for a fixed period of time but may be allowed access to multiple storage devices governed by the KDC within that period.

III. PROPOSED SYSTEM

pNFS separates the file system protocol processing into two parts: metadata processing and data processing. Metadata is information about a file system object, such as its name, location within the namespace, owner, permissions and other attributes.

The entity that manages metadata is called a metadata server. On the other hand, regular files data is striped and storedacross storage devices or servers. Data striping occurs in at least two ways: on a file-by-file basis and, within sufficiently large files, on a block-by-block basis. Unlike NFS, a read or write of data managed with pNFS is a direct operation between a client node and the storage system itself.

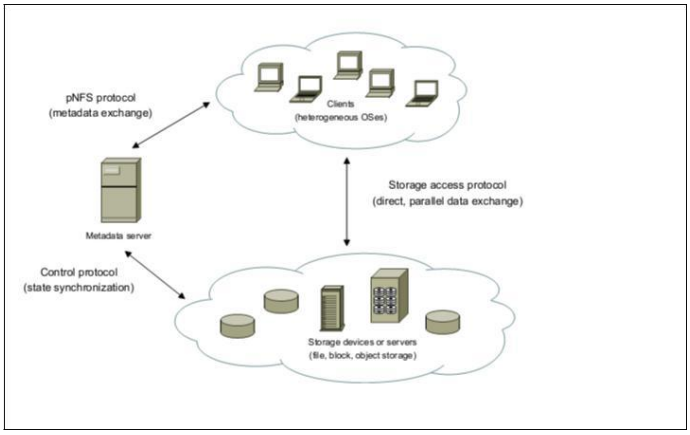

Fig. 1. Proposed System

pNFS having the collection of three protocol

i) pNFS Protocol- The pNFS protocol is work between the client and metadata server. It is used for metadata exchange.

@IJAERD-2016, All rights Reserved

ii) Storage Access Protocol- The Storage Access Protocol is work between the client and storage device.The Storage

Access Protocol that specifies how the client is access data from particular storage device.

(iii) Control Protocol- The Control Protocol is work between the metadata server and storage device.Thec Control protocol that synchronizes state between the metadata server and the storage devices.

IV. OVERVIEW OF OUR PROTOCOL

In these protocols,we focus on parallel session key es-tablishment between a client and n different storage devices through a metadata server. It provides the many-to-many communications between clients and storage devices.

A. Design Goals

In this system, we focus on efficiency and scalability with respect to the metadata server. That is, our goal is to reduce the workload of the metadata server. On the other hand, the computational and communication overhead for both the client and the storage device should remain low.In fact, we consider a stronger security model with forward secrecy for three of our protocols.we would like an escrow-free solution, that is, the metadata server does not learn the session key shared between a client and a storage device.

B. Main idea

We let M denote a metadata server,C denote a client, and S denote a storage device. Let entity X,Y " fM,C,S g we then use ID_x to denote a unique identity of X, and K_x to denote Xs secret (symmetric) key; while K_{xy} denotes a secret key shared between X and Y, and s k denotes a session key. we let E(K;m) be a standard (encryption only) symmetric key encryption function. where both functions take as input a key K and a message m. Finally, we use t to represent a current time and to denote a layout. We may introduce other notation as required.

pNFS-AKE-I: Our first protocol can be regarded as a modified version of Kerberos that allows the client to generate its own session keys. That is, the key material

used to derive a session key is pre-computed by the client for each v and forwarded to the corresponding storage device in the form of an authentication token at time t (within v). As with Kerberos, symmetric key encryption is used to protect the confidentiality of secret information used in the protocol. However, the protocol does not provide any forward secrecy. Further, the key escrow issue persists here since the authentication tokens containing key materials for computing session keys are generated by the server.

pNFS-AKE-II: To address key escrow while achieving forward secrecy simultaneously, we incorporate a Diffie-Hellman key agreement technique into Kerberos-like
pNFS-AKE-III: Our third protocol aims to achieve full forward secrecy, that is, exposure of a long-term key

affects only a current session key (with respect tot), but not all the other past session keys. We would also like to prevent key escrow. In a nutshell, we enhance pNFS-AKE-II with a key update technique based on any efficient one-way function, such as a keyed hash function. In Phase I, we require C and each Si to share some initial key material in the form of a Diffie-Hellman key. In Phase II,the initial shared key is then used to derive session keys in the form of a keyed hash chain. Since a hash value in the chain does not reveal information about its pre-image, the associated session key is forward secure.

V. ALGORITHMIC STRATEGY

RSA is an algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of them can be given to everyone. The other key must be kept private. It is based on the fact that finding the factors of an integer is hard (the factoring problem). RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman, who first publicly described it in 1978. A user of RSA creates and then publishes the product of two large prime numbers, along with an auxiliary value, as their public key. The prime factors must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime factors can feasibly decode the message.

@IJAERD-2016, All rights Reserved

1. Choose two different large random prime number p and q
2. Calculate n=pq

   n is the modulas for the public key the private key.

3. Calculate the : (n)=(p-1)(q-1)

4. Choose an integer e such that 1 < e < (n) and e is co-prime to (n) that is e and (n) share no factors other than 1.

   gcd(e, (n))=1.

   e is released as the public key exponent.

5. Compute d to satisfy the congruence relation de=1(mod (n)) i.e. de=1+k* (n). for k is some integer number. d is kept as the private key exponent. choose d to satisfy de=1(mod (n)) The public key is,for a padded message m the encryption function is

   $C=m^e \bmod n$

   The private key is the decryption. $m=c^d \bmod n$

VI. ADVANTAGES

1) This system provides time efficient and very effective secure file sharing.
2) It is increase the scalability in network.
3) It is also provide the forward secrecy and escrow free mechanism.
4) It minimize the workload of metadata server.

VII. CONCLUSION AND FUTURE WORK

In our system key exchange protocol are for parallel net-work file system.this protocol having advantages over the exiting system.the three protocol are provide scalability,escrow free,forward secrecy.protocol is used to reduce the overload metadata server.and it also transfer the layout between meta-data and client node and secure communication take place between client and server.no of clients node access the no of server parallel.using pNFS protocol high performance and efficiency is achieved.The data loss is achieved through fault tolerant stripping algorithm. Using RSA algorithm key ex-change take place.

REFERENCES

[1] IEEE Transactions on Parallel and Distributed Systems VOL. X, NO. XX, 2015 Authenticated Key Exchange Protocols for Parallel,Hoon Wei Lim ,Guomin Yang .
[2] S. Shepler, M. Eisler, and D. Noveck. Network file system (NFS) version 4 minor version 1 protocol.The Internet Engineering Task Force (IETF), RFC 5661, Jan 2012.
[3] J. Linn. The Kerberos version 5 GSS-API mechanism. The Internet Engineering Task Force (IETF), RFC 1964, Jun 2011.
[4] B. Welch, M. Unangst, Z. Abbasi, G.A. Gibson, B. Mueller, J. Small, J. Zelenka, and B. Zhou. Scalable performance of the Panasas parallel file system. InProceedings of the 6th Conference on Storage devices and files Technologies (FAST), pages 1733. USENIX Association, Feb 2011.
[5] B.S. White, M. Walker, M. Humphrey, and A.S. Grimshaw. Le-gionFS: A secure and scalable file system supporting cross-domain high-performance applications. InProceedings of the IEEE Conference paper on computing (C), page 59. ACM Press, Nov 2011.
[6] M. Abd-El-Malek, W.V. Courtright II, C. Cranor, G.R. Ganger, J. Hen-dricks, A.J. Klosterman, M.P. Mesnier, M. Prasad, B. Salmon, R.R. Sam-basivan, S. Sinnamohideen, J.D. Strunk, E. Thereska, M. Wachs, and J.J. Wylie. Ursa Minor: Versatile cluster-based storage. In Proceedings of the 4th USENIX Conference on File and Storage Technologies (FAST), pages 5972. USENIX Association, Dec 2010.
[7] Y. Zhu and Y. Hu. SNARE: A strong security scheme for network-attached storage. InProceedings of the 22nd Symposium on Reliable Distributed Systems (SRDS), pages 250259. IEEE Computer Society, Oct 2009.

@IJAERD-2016, All rights Reserved