International Journal of Engineering Research-Online
A Peer Reviewed International Journal
Email:editorijoer@gmail.com http://www.ijoer.in

Vol.4., Issue.2., 2016 (Mar-Apr)

RESEARCH ARTICLE

ISSN: 2321-7758

SECURE ACCESS FILES BASED ON SESSION KEYS USING AUTHENTICATED KEY EXCHANGE PROTOCOL

SHARVINI N S[1], VIMALA S[2]
[1]Department of MCA, Panimalar Engineering College, CHENNAI
[2]Professor, Department of MCA, Panimalar Engineering College, CHENNAI

ABSTRACT
In data communication system Key establishment is the major process for providing security to the entire network while transfer the data between neighbors. We study the Problem of "Authenticated Key Exchange Protocol For Parallel Network File System" provide that the meta data server to serve that the client and storage server to response the client user and meta data server will response to all the user's request and will accept the user request based on the their secrecy type and will provide the approval and then user need to select the files as the secrecy type. In existing process the user can access only one files to the secrecy type and one of the kerberos based PNFS protocol will use to handle that the workload and user misuse the sharing files easily. Meta data server manually generate the key and it would be send to the user while sending the key itself meta data server would create the session time problem will be occur within the particular time user can use their file from the server. The proposed system will overcome all the problem and user can access the multiple files and two types of secrecy will use to avoid the workload process within that timing and to authenticate the key exchange protocol after session time got over, the session time user have to give the request again to the meta data server.

Keywords—Authenticated key exchange, network file systems, fully forward secrecy, partially forward secrecy, Escrow-free.

I INTRODUCTION

In Authenticated Key Exchange Protocol For Parallel Network File System, the Communication Between Clients and Server is very secure Data Transaction. The User's can access the data with the permission of the server and secure method. The server will accept the user request based on their secrecy type and server response to the clients. In the data transaction there is a chance for file missing and misuse the sharing of files to prevent this problem, we proposed this system after the investigation from user's feedback report.

We developed this System Application with set protocol for an intuitive method to alleviate this problem .The server will provide the approval to user's Meta data(meta data means data about data) server is manually generate the Key. The generate key send them to User's and they can access the data by using of secrecy key. Our primary goal in this work is to design efficient andsecure authenticated key exchange protocols that meet specific requirements of pNFS.

- *Scalability* – the metadata server facilitating access requests from a client to multiple storage devices should bear as little workload as possible such that the server will not become a performance bottleneck, but is capable of supporting a very large number of clients;
- *Forward secrecy* – the protocol should guarantee the security of past session keys when the long-term secret key of a client or a storage device.
- *Escrow-free* – the metadata server should not learn any information about any session key used by the client and the storage device, provided there is no collusion among them.

In this Application, the secured data transaction between clients and server, the user's no chance to access the data without the permission from the server. The user's must be only access the data by the authentication key and users can access multiple files and two ways of secrecy will use to avoid the workload process within that time and authentication method is present to key exchange protocol. The main results of this paper are three new provably secure authenticated key exchange protocols.

Our protocols, progressively designed to achieve each of the above properties, demonstrate the trade-offs between efficiency and security. We show that our protocols can reduce the workload of the metadata server by approximately half compared to the current Kerberos-based protocol, while achieving the desired security properties and keeping the computational overhead at the clients and the storage devices at a reasonably low level.The user can work with their approval files. In the transaction and allocation time no misuse the key by other user's. From this system, user's can satisfy in data transaction between the clients and server.

II EXISTING SYSTEM

A.EXISTING CONCEPT

In existing protocol, kerberos based PNFS protocol, therethe Meta data server will encrypt files and will upload that into the cloud, and will create the instance name to the every file, and there will be many workload while send the key to the user and also PNFS will not provide any secrecy to the file.

On the other hand, data grids and file systems such as, Ocean Store, LegionFS and FARSITE, make use of public key cryptographic techniques and public key infrastructure (PKI) to perform cross-domain user authentication.

B.DRAWBACKS

- File request delay for the client user and easily take over the time.
- Server sharing the key to misuse the other user easily handle them.
- Client side work load will affect.

III PROPOSED SYSTEM

A.PROPOSED CONCEPT

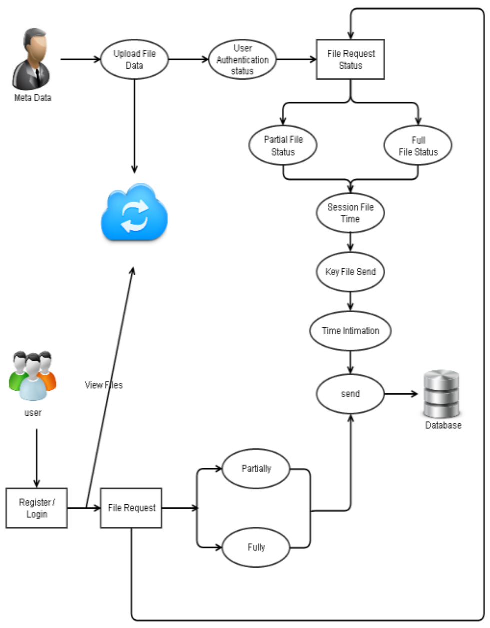
In this work, we investigate the problem of secure many to- many communications in large-scale network file systems that support parallel access to multiple storage devices.

In proposed system we have two secrecy type to overcome the all existing problem and Partial forward secrecy and fully forward secrecy, the both secrecy will give the security to the shared files and also user cannot share the key to the other user's as well..

B.ADVANTAGES

- Multiple files Access to the user.
- Time Allocation and do not misuse key type to the other users.
- There are Partial protocol and various protocol will work and to avoid the workload.

IV SYSTEM ARCHITECTURE



V MODULE DESCRIPTION

A. Add files and accepting the Request

In this module meta data server would add the all the data into the server, and all the added files would be visible to the all the user's, but they cannot access it to directly so user would select the secrecy type and will the request to the accessing request to the meta data server and then meta data server will accept the request and give the permission to the user.

B. View files and accepting the Request

In this module meta data server will watch all the user's request and will accept the user request based on the their secrecy type and will provide the approval and then user need to select the files first the secrecy type , after that only they can give the request.

C. Send the Key and accepting the File

After approving the request will the send the accessing key to the user. Meta data server manually generate the key and the would be send to the user while sending the key itself meta data server would create the session time within the particular time user can use their use their file from the server.

D. Partially Forward Secrecy

In this module user can select the multiple file in that particular session time .within that timing user can work with their approval files. User cannot access the other than the approval files. And also user should give the accessing key to access that file .otherwise user cannot access their file. After the session time file will be automatically will come outside .

E. Fully Forward Secrecy

In this module user can select the only one file in that particular session time .within that timing user can work with their approval files. User cannot access the other than the approval files. And also user should give the accessing key to access that file .otherwise user cannot access their file. After the session time file will be automatically will come outside.

F. Get Mac Address

In this module according to their secrecy type they can access their authorized files,and while accessing the file automatically will get the user mac address will forward it in to the admin , so meta data server easy can identify the problem if there is any misuse or any irrelevant modification, admin can easy can identify the user.

VI DESCRIPTION OF OUR PROTOCOLS

We first introduce some notation required for our protocols.Let$F(k;m)$ denote a secure key derivation function that takesas input a secret key k and some auxiliary information m,and outputs another key.

Let *sid*denote a session identifier which can be used to uniquely name the ensuing session. Let also N be the total number of storage devices to which a client is allowed to access. We are now ready to describe the construction of our protocols..

pNFS-AKE-I

Our first pNFS-AKE protocol is illustrated in Figure 2. Foreach validity period v, C must first pre-compute a set of key materials*KCS*1 ; : : : ;*KCSN* before it can access any of the*N* storage device *Si* (for 1 _ *i* _ *N*). The key materials aretransmitted to *M*.

Phase I – For each validity period *v*:

(1) *C !M* : *IDC*, *E*(*KCM;KCS*1 *; : : : ;KCSN*)

(2) *M !C* : *E*(*KMS*1 ; *IDC; IDS*1 *; v;KCS*1 *); : : : ; E*(*KMSN* ; *IDC; IDSN ; v;KCSN*)

Phase II – For each access request at time *t*:

(1) *C !M* : *IDC, IDS*1 *; : : : ; IDSn*

(2) *M !C* : _1*; : : : ; _n*

(3) *C !Si* : *_i; E*(*KMSi; IDC; IDSi ; v;KCSi*), *E*(*sk*0 *i* ; *IDC; t*)

(4) *Si !C* : *E*(*sk*0 *i* ; *t* + 1)

Fig. 2.Specification of pNFS-AKE-I.

pNFS-AKE-II

We now employ a Diffie-Hellman key agreement techniqueto both provide forward secrecy and prevent key escrow. In this protocol, each *Si* is required to pre-distribute some keymaterial to *M* at Phase I of the protocol. *Si* must verify the authentication token to ensure the integrity of *gc*. Here *C* and *Si* compute *skzi* for *z* = 0*;* 1 as follow:

skz i = F(gcsi; IDC; IDSi ; gc; gsi ; v; sid; z):

At the end of the protocol, *C* and *Si* share a session key*sk*1*i* .

pNFS-AKE-III

As explained before, pNFS-AKE-II achieves only partial forward secrecy (with respect to *v*). In the third variant of our pNFS-AKE, therefore, we attempt to design a protocol that achieves full forward secrecy and escrow-freeness. A straight forward and well-known technique to do this is through requiring both *C* and *Si* to generate and exchange fresh Diffie-Hellman components for each access request at time *t*. However, this would drastically increase the computational overhead at the client and the storage devices. Hence, we adopt a different approach here by combining the Diffie-Hellman key exchange technique used in pNFS-AKE-II with a very efficient key update mechanism.

VII OTHER RELATED WORK

Some of the earliest work in securing large-scale distributed file systems, have already employed Kerberos for performing authentication and enforcing access control. Kerberos, being based on mostly symmetric key techniques in its early deployment, was generally believed to be more suitable for rather closed, well-connected distributed environments. Each user of these systems is assumed to possess a certified public/private key pair. However, these systems were not designed specifically with scalability and parallel access in mind. Nevertheless, these proposals assumed that a metadata server shares a group secret key with each distributed storage device. The group key is used to produce capabilities in the form of message authentication codes. However, compromise of the metadata server or any storage device allows the adversary to impersonate the server to any other entities in the file system. This issue can be alleviated by requiring that each storage device shares a different secret key with the metadata server. Nevertheless, such an approach restricts a capability to authorising I/O on only a single device, rather than larger groups of blocks or objects which may reside on multiple storage devices.

VIII CONCLUSION

The based PNFS protocol will use to avoid that the workload. After the session time, file will be automatically come outside till the session time getting exceeded automatically, intimation will come for timing session if user wants that extra time .User can use that extra time or else that page would close automatically after session time got over, after the session time user want to give the request again to the meta data server. We proposed three authenticated key exchange protocols for parallel network file system (pNFS). The three appealing advantages are offered by our protocols over the existing Kerberos-based pNFS protocol. Firstly the metadata server executing our protocols has much lower workload as compared to that Kerberos-based approach. Secondly, two our protocols provide forward secrecy:one which is partially forward secure (with respect to multiple sessions within a time period), while the other is fully forward secure (with respect to a session). Thirdly, we also have designed a protocol which provides forward secrecy as well as is escrow-free.

IX REFERENCES

[1]. M. Eisler. LIPKEY - A Low Infrastructure Public Key mechanism using SPKM. *The Internet Engineering Task Force (IETF)*, RFC 2847, Jun 2000.

[2]. M. Eisler. XDR: External data representation standard. *The Internet Engineering Task Force (IETF)*, STD 67, RFC 4506, May 2006.

[3]. M. Eisler. RPCSEC GSS version 2.*The Internet Engineering Task Force (IETF)*, RFC 5403, Feb 2009.

[4]. M. Eisler, A. Chiu, and L. Ling. RPCSEC GSS protocol specification. *The Internet Engineering Task Force (IETF)*, RFC 2203, Sep 1997.

[5]. S. Emery. Kerberos version 5 Generic Security Service Application Program Interface (GSS-API) channel binding hash agility.*The InternetEngineering Task Force (IETF)*, RFC 6542, Mar 2012.

[6]. M. Factor, D. Nagle, D. Naor, E. Riedel, and J. Satran. The OSD security protocol. In *Proceedings of the 3rd IEEE International Securityin Storage Workshop (SISW)*, pages 29–39. IEEE Computer Society, Dec 2005.

[7]. http://www.fsgrid.com/ Financial Services Grid Initiative.

[8]. S. Ghemawat, H. Gobioff, and S. Leung. The Google file system. In *Proceedings of the 19th ACM Symposium on Operating SystemsPrinciples (SOSP)*, pages 29–43. ACM Press, Oct 2003.

[9]. G.A. Gibson, D.F. Nagle, K. Amiri, J. Butler, F.W. Chang, H. Gobioff, C. Hardin, E. Riedel, D. Rochberg, and J. Zelenka. A costeffective, high-bandwidth storage architecture.*ACM SIGPLAN Notices*, 33(11):92–103. ACM Press, Nov 1998.

[10]. http://wiki.apache.org/hadoop/PoweredBy

[11]. J.H. Howard, M.L. Kazar, S.G. Menees, D.A. Nichols, M. Satyanarayanan, R.N. Sidebotham, and M.J. West. Scale and performance in a distributed file system. *ACM Transactions on Computer Systems(TOCS)*, 6(1):51–81. ACM Press, Feb 1988.

[12]. F. Hupfeld, T. Cortes, B. Kolbeck, J. Stender, E. Focht, M. Hess, J. Malo, J. Marti, and E. Cesario. The XtreemFS architecture – a case for objectbased file systems in grids. *Concurrency and Computation: Practice andExperience (CCPE)*, 20(17):2049–2060. Wiley, Dec 2008.

[13]. B.S. White, M. Walker, M. Humphrey, and A.S. Grimshaw.LegionFS: A secure and scalable file system supporting cross-domain highperformance applications. In *Proceedings of the ACM/IEEE Conferenceon Supercomputing (SC)*, page 59. ACM Press, Nov 2001.

[14]. J. Kubiatowicz, D. Bindel, Y. Chen, S.E. Czerwinski, P.R. Eaton, D. Geels, R. Gummadi, S.C. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B.Y. Zhao. OceanStore: An architecture for global-scale persistent storage. In *Proceedings of the 9th International Conferenceon Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 190–201. ACM Press, Nov 2000.

[15]. S. Langella, S. Hastings, S. Oster, T. Pan, A. Sharma, J. Permar, D. Ervin, B.B. Cambazoglu, T.M. Kurc,, and J.H. Saltz. Model formulation:Sharing data and analytical resources securely in a biomedical research grid environment. *Journal of the American Medical InformaticsAssociation (JAMIA)*, 15(3):363–373. BMJ, May 2008.

[16]. A.W. Leung and E.L. Miller. Scalable security for large, high performance storage systems. In *Proceedings of the ACM Workshop on StorageSecurity and Survivability (StorageSS)*, pages 29–40. ACM Press, Oct 2006.

[17]. A.W. Leung, E.L. Miller, and S. Jones. Scalable security for petascale parallel file

systems. In *Proceedings of the ACM/IEEE Conference onHigh Performance Networking and Computing (SC)*, page 16. ACM Press, Nov 2007.

[18]. H.W. Lim. Key management for large-scale distributed storage systems. In *Proceedings of the 6th European Public Key Infrastructure Workshop(EuroPKI)*, pages 99–113. Springer LNCS 6391, Sep 2010.

[19]. J. Linn. The Kerberos version 5 GSS-API mechanism. *The Internet Engineering Task Force (IETF)*, RFC 1964, Jun 1996.

[20]. S.A. Weil, S.A. Brandt, E.L. Miller, D.D.E. Long, and C. Maltzahn. Ceph: A scalable, high-performance distributed file system. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI)*, pages 307–320. USENIX Association, Nov 2006.

[21]. http://www.chennaisunday.com/2015DOTNET/Authenticated%20Key%20Exchange%20Protocols.pdf

[22]. http://www.slideshare.net/optimusproject/authenticated-key-exchange-protocols-for-parallel-network-file-systems-58963129