International Journal of Scientific Research and Engineering Development-– Volume 6 Issue 4, July- Aug 2023

Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Parallel Network File Systems Using Authendicated Key Exchange Protocols

M.Mohamed Faisal*, K.Vignesh**

*(Asst Professor CSE, SembodaiRukmaniVaratharajan Engineering College, and Sembodai
Email: faisalit85@gmail.com)

** (PG Scholar CSE, SembodaiRukmaniVaratharajan Engineering College, and Sembodai
Email: dhivavignesh@gmail.com)

----------**********----------

Abstract:

We study the problem of key establishing for secure many-to-many communications. The problem is inspired by the proliferation of large-scale distributed file systems supporting equivalent access to multiple storage devices. Our work submissions on the current Internet standard for such file systems, i.e., parallel Link File System which makes use of Kerberos to establish parallel session keys between customers and storage devices.

Our analysis of the existing Kerberos-based protocol shows that it has a number of limitations:

(i) A metadata server facilitating key exchange between the customers and the storage devices has heavy workload that restricts the scalability of the protocol.

(ii) The protocol does not provide onward secrecy.

(iii) The metadata waiter generates itself all the session keys that are used between the clients and packing devices, and this inherently leads to key escrow.

In this paper, we propose a variety of authentic key exchange protocols that are designed to address the above issues. We show that our proprieties are capable of reducing up to almost 54% of the capacity of the metadata server and concurrently supporting onward secrecy and escrow-freeness. All this requires only a small section of increased computation overhead at the client.

Keywords **—Communication,System,Customers,storage,device.**

----------**********----------

I.INTRODUCTION

In a parallel file system, file data is distributed across multiple packing devices or nodes to allow concurrent access by multiple tasks of a parallel application. This is typically used in large-scale cluster calculating that focuses on high performance and reliable access to big datasets. That is, higher I/O bandwidth is achieved through concurrent access to multiple storing devices within large compute clusters; while data loss is protected

through data mirroring using fault-tolerant striping algorithms. Some examples of high presentation parallel file systems that are in construction use are the IBM General Parallel File System (GPFS), Google File System (Google FS), Lustre, Parallel Virtual File System (PVFS), and Panasas File System; while there also exist research projects on distributed object storage systems such as Usra Minor, Ceph, Xtreem FS, and G farm. These are usually required for advanced scientific or data-intensive applications such as, seismic data processing, digital animation studios, computational fluid dynamics, and semiconductor manufacturing. In these environments, hundreds or thousands of file system clients share data and create very high aggregate I/O load on the file system supporting peta byte- or terabyte-scale packing capacities.

Independent of the development of cluster and high performance computing, the appearance of clouds and the Map Reduce programming model has resulted in file systems such as the Hadoop Scattered File System (HDFS), Amazon S3 File System, and Cloud Store. This, in turn, has accelerated the wide-spread use of disseminated and parallel computation on big datasets in many association. Some notable users of the HDFS include AOL, Apple, eBay, Face book, Hewlett-Packard, IBM, LinkedIn, Twitter, and Yahoo!.

In this work, we investigate the problem of secure many to many communications fashionable big-scale network file organizations that support parallel access to various packing devices. That is, we consider a communication model where there are a large quantity of clients (potentially hundreds or thousands) accessing multiple remote and distributed storage devices (which also may scale up to hundreds or thousands) in parallel. Particularly, we focus on how to interchange key materials and create parallel secure conferences between the clients and the storage devices in the parallel Link File System the current Internet standard—in an efficient and scalable manner. The development of is driven by Panasas, Netapp, Sun, EMC, IBM, and CITI, and thus it shares many common structures and is compatible with many existing commercial/recorded network file systems.

II.LITERATURE SURVEY

2.1 GENERIC SECURITY SERVICE APPLICATION PROGRAM INTERFACE (GSS-API)

S. Emery is discussed about the network bindings are implemented using an MD5 hash in the Kerberos Version 5 Generic Security Service Application Programming Interface (GSS-API) mechanism. This updates to allow network bindings spending algorithms exchanged based on Kerberos crypto framework.

In addition, because this update makes use of the last extensible field in the Kerberos client-server exchange message, allowances are defined to allow future protocol extensions

With the recently discovered weaknesses in the MD5 hash algorithm, there is a need to use sturdier hash algorithms. Programming Interface (GSS-API) instrument uses MD5 towards calculate frequency binding verifiers. This document specifies inform to the mechanism that allows it to create channel binding data based on exchanged algorithms. This will agree deploying new algorithms incrementally without breach interoperability with older implementations when new attacks arise in the future.

2.2 COLLUSION RESISTANT BROADCAST ENCRYPTION WITH SHORT CIPHERTEXTS AND PRIVATE KEYS. IN ADVANCES IN CRYPTOLOGY – PROCEEDINGS OF CRYPTO

D. Boneh, C. Gentry, and B. Waters described two new community key broadcast encryption systems for nationless receivers. Both systems are fully safe against any number of colluders. In their first structure both cipher texts and reserved keys are of constant size for any subset of receivers. The public key size in this system is line in the total number of receivers. Second system is a generalization of the first that provides a trade-off between cryptograph text size and public key size. For example, we achieve a collusion resistant broadcast system for n users where both cipher texts and public keys are of size for any subset of telephones. We discuss several submissions of these systems.

No matter what the receiver set is, the broadcast cipher text contains only two group elements. Each user's remote key is just a solo group element. Thus, for example, when broadcasting to small sets our system is far better than the trivial solution discussed above. This is not a large problem in applications such as encrypted file systems where the receivers have access to a large shared storage middle in which the public key can be stored.

III.SYSTEM ANALYSIS

System analysis is the overall analysis of the system before implementation and for incoming at a precise solution. Careful analysis of a system before application prevents post implementation problems that capacity arise due to bad analysis of the problem statement.

Thus the necessity for systems analysis is justified. Analysis is the first crucial step, detailed study of the various operations achieved by a system and their relationships within and outside of the system. Analysis is defining the borders of the system that will be followed by design and implementation.

3.1 EXISTING SYSTEM

- The problem is encouraged by the large-scale distributed file systems supporting parallel access to multiple storage devices.

- Our work focuses on the current Internet standard for such file systems, i.e., parallel Network File System .which makes use of Kerberos to establish parallel session keys between clients and storage devices. Our review of the standing Kerberos-based protocol shows that it has a figure of limitations: metadata server facilitating key exchange between the clients and the storage devices has heavy workload that restricts the scalability of the protocol; the protocol does not provide forward secrecy; the metadata server generates itself all the assembly keys that are used between the consumers and storage devices, and this fundamentally leads to key escrow.

3.2 PROPOSED SYSTEM

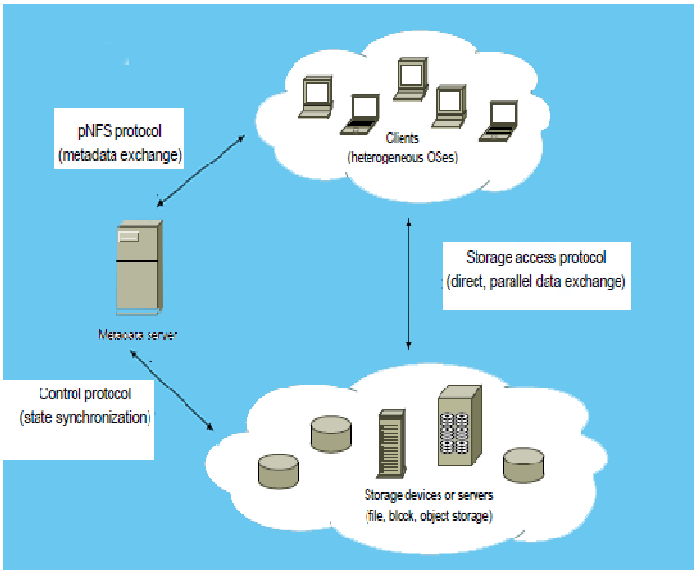
We show that our protocols are capable of reducing up to approximately of the workload of the metadata server and concurrently supporting forward secrecy and escrow-freeness. All this entails only a small fraction of improved computation overhead at the client.

- Proposed System Algorithms
- Fault-tolerant striping algorithms.

3.2.1 Advantages

Finally, in the last arugmented game, we can right that the adversary has no advantage in charming the game since a random key is returned to the enemy. Our protocols offer three appealing advantages over the existing Kerberos-based protocol.

IV. ARCHITECTURE DIAGRAM



V. SYSTEM MODULES

5.1 SYSTEM MODULES

This project is entitled as Authenticated Key Exchange Protocols for Parallel Network File System using Java as Front end and SQL Server as back end.

- User Registration and Login

- Generation of Keys
- File Uploading
- Verification of file and User
- Stored in cloud
- Performance Evaluation

5.2 MODULE DESCRIPTION

5.2.1 User Registration and Login

- In Initialization of User, user enter the personal information to register into application.
- After register the information, user can login into the process through the username and password.

5.2.2 Key Generation

- Secret key experience can happen in this system model.
- The main results of this newspaper are three new provably secure authenticated key exchange protocols.
- We describe our design goals and give some awareness of a variety of authenticated key exchanged protocols that we consider in this work.

VI. CONCLUSION

We proposed three protocols for parallel link file system protocols offer three interesting advantages over the existing protocol. First, the metadata server executing our protocols has much lower workload than that of the Kerberos-based approach. Second, two our protocols provide forward secrecy: one 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). Third, we have designed a protocol which not only provides forward secrecy, but is also escrow-free.

REFERENCES

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

[2] D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Advances in Cryptology – Proceedings of CRYPTO, pages 258–275. Springer LNCS 3621, Aug 2015.

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

[4] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. Communications of the ACM, 53(4):50–58. ACM Press, Apr 2010.

[5] 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 and Experience (CCPE), 20(17):2049–2060. Wiley, Dec 2008

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

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

[8] R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Advances in Cryptology – Proceedings of EUROCRYPT, pages 453–474. Springer LNCS 2045, May 2001.

[9] Crypto++ 5.6.0 Benchmarks. http://www.cryptopp.com/benchmarks. html.

[10] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. In Proceedings of the 6th Symposium on Operating System Design and Implementation (OSDI), pages 137–150. USENIX Association, Dec 2004.

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

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

