INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 2 ISSUE 5 (2016) PAGES 1262 - 1268
RECEIVED : 21/10/2016. PUBLISHED : 27/10/2016
October 27, 2016

IMPROVE THE FORWARD SECRECY IN PARALLEL NETWORK FILE SYSTEM WHILE MINIMIZING THE WORK LOAD OF METADATA SERVER

[1]Vincy Jeba Shanthi.S, [2]Parameswari.V,
[1]Research Scholar, Department of Computer Science, Bharathiyar Arts and Science College for Women's, Deviyakurichi, Salem.
[2]Assistant Professor, Department of Computer Science, Bharathiyar Arts and Science College for Women's, Deviyakurichi, Salem.

Abstract:
Key establishment is the main issue for secure many-to-many communications. We study the problem of large-scale distributed file system, which support parallel access to multiple storage devices. The existing Kerberos-based protocol has a number of limitations: scalability of the protocol is restricted, it provide less forward secrecy, and leads to key escrow. In this paper, we propose a variety of authenticated and encrypted key exchange protocols that are designed to address the above issues. The proposed protocol for parallel network file system is AE-KEP-I and AE-KEP-II (Authenticated and Encrypted key exchange protocol). Compare to the existing Kerberos based protocols our proposed protocols are significantly reduce the workload of the metadata server and concurrently supporting forward secrecy and escrow-freeness.

Index Terms— Parallel sessions, Authenticated and Encrypted key exchange, Network File Systems, Metadata server, Forward secrecy and key escrow.

1. INTRODUCTION

The Parallel file system and Network File System are mainly used to build a high-performance computing (HPC) cluster. NFS is generally considered easy to use but the FS does not scale well across large clusters. In parallel file systems, a few nodes connected to the storage and the large files are shared across multiple nodes, the main advantage of a parallel file system can grant a global name space, scalability [10]. A parallel file system includes a metadata server (MDS), Metadata is the information about a file for example, its name, location, and owner. How are Parallel File Systems different from Distributed File Systems? Distributed file systems often store entire objects (files) on a single storage node. Parallel file systems allocate data of a single object across multiple storage nodes [15]. Distributed file systems frequently run on architectures where the storage area is co-located with the application. *Fault--tolerance:* Distributed file systems take on fault--tolerance responsibilities but the Parallel file systems run on enterprise shared storage [7]. *Workloads:* Distributed file systems are geared for loosely coupled, distributed applications. Parallel file systems target HPC applications, which tend to perform highly coordinated I/O accesses, and have massive bandwidth requirements. In this work, we study the problem of secure many- to-many communications in large-scale network file systems (NFSs) that carry parallel access to numerous storage devices [9],[3]. That is, we consider a communication model where there are a huge number of clients (hundreds or thousands) accessing multiple remote and distributed storage devices (which furthermore scale up to hundreds or thousands) in parallel. The key exchange problem [1] is how to exchange whatever keys or other information are needed so that no one else can obtain a copy [11].

INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 2 ISSUE 5 (2016) PAGES 1262 - 1268
RECEIVED : 21/10/2016. PUBLISHED : 27/10/2016
October 27, 2016

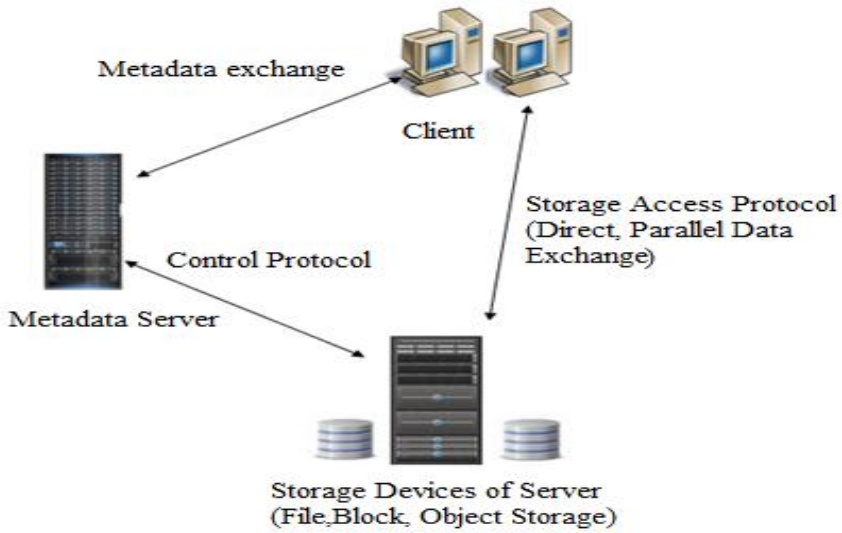

Fig.1. The conceptual model of pNFS

With the advent of public key and private key cipher algorithms, the encrypting key could be made in public scenario, while no one without the knowledge of decrypting key. Public key infrastructures (PKIs) have been proposed as a solution for this problem of identity authentication [7], [13]. In their most usual implementation, each user requests a digital certificate from a "certificate authority" which serves for added clients as a non-tamper able authentication of identity; at the risk of compromising every legitimate client in case the CA [1] itself is compromised. Key-exchange protocols (KE) are mechanisms by which two parties that communicate over an adversarial controlled network can generate a common secret key [5]. KE protocols are essential for enabling the use of shared-key cryptography to protect transmitted data over insecure networks [4], [13]. As such they are a central piece for building secure connection, and are the most commonly used cryptographic protocols.

2. PREVIOUS WORK

Previous work focuses on the parallel Network File System, which makes use of Kerberos to establish parallel session keys between clients and storage devices [4]. Kerberos is a computer network authentication protocol that works on the basis of 'tickets' to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner. It increased computation overhead at the client side. Review of the existing Kerberos-based protocol shows that it has a number of limitations [12]. The existing protocol contains the following limitations: (i). A metadata server facilitating key exchange among the clients and the storage devices has large or heavy workload that restricts the scalability of the protocol. (ii).The protocol does not support forward secrecy. (iii).The metadata server generates all the session keys that are used among the clients and storage devices, and this naturally leads to key escrow [1].

3. PROBLEM FORMULATION

In this paper, we propose a variety of authenticated and encrypted key exchange protocols that are designed to overcome the above problems. The proposed protocol for parallel network file system is AE-KEP-I and AE-KEP-II (Authenticated and Encrypted key exchange protocol). The proposed work overcomes the limitations of the existing work with the help of Authenticated and Encrypted key exchange protocols. (i). Scalability of the proposed protocols AE-KEP-I and AE-KEP-II have been

increased. (ii). It avoids the key escrow and other security problems. It also provides provide forward secrecy. (iii). Compare to existing protocol the proposed protocols reduces the workload of the metadata server.

4. RELATED WORK

A. J. Klosterman, C. Cranor, M. P. Mesnier, J. Hendricks, and G. R. Ganger proposed a concep of cluster based storage system in *Ursa [2]*. Ursa Minor achieves its versatility by using a protocol family unit, storing variably sized data fragments at individual storage nodes, and maintaining per-object data distribution descriptions. It supports a per object choice of data distribution. Cluster-based storage [2] has cost advantages and scalability. W. J. Bolosky, R. Chaiken, M. Castro, G. Cermak, and A .Adya, discus about a concept of scalable file system in *FARSITE [3]*. It is a secure, scalable file system that logically functions as a centralized file server [3] but is physically distributed among a set of un-trusted computer. Farsite provides file availability and reliability through randomized replicated storage. It may have performance problems; it does not provide data privacy from users who have physical access to the directory group members.

M. Lillibridge, M. Ji, M. K. Aguilera, and J. MacCormick, illustrate a cryptography concept in *"Block-Level Security" [4].* It enforces security using the well-known idea of self-describing capabilities, with two novel features that limit the need for memory on secure NADs [4], which is a scheme to manage revocations based on capability groups. Network-attached disk can be used to make file systems that provide better performance. When a network partition separates the metadata server from a disk, the server is unable to revoke capabilities for thatdisk,

E. L. Miller, and A. W. Leung, describes a concept in *"Scalable security storage systems" [5]*. It aims to propose a scalable and efficient protocol for safety measures in high performance and object based storage systems. To reduce the protocol overhead [5] and eliminate bottlenecks to increase performances without sacrificing security primitives.

Y. Chen, D. Bindel and J. Kubiatowicz, discus about a concept of large scale storage in *"OceanStore" [6]*. It is an effectiveness infrastructure designed to span the globe and offer continuous access to constant information. To improve performance, data is tolerable to be cached anywhere, anytime. In addition, monitoring of usage patterns [6] allows adaptation to regional outages and denial of service attacks.

B. Kolbeck, Hupfeld, and T. Cortes, illustrate the concept of Grid management in *"XtreemFS" [10].* It analyzes the predominant approach and argues that object-based file systems can be an alternative when adapted to the characteristics of a Grid environment [10].

5. AE-KEP: AUTHENTICATED AND ENCRYPTED KEY EXCHANGE PROTOCOLS

In these protocols, we focus on establishment of parallel session key between a client and *n* different storage devices through a metadata server. It can be extended straightforwardly to the multi-user setting, i.e., many-to-many communications between clients and storage devices.

INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 2 ISSUE 5 (2016) PAGES 1262 - 1268
RECEIVED : 21/10/2016. PUBLISHED : 27/10/2016
October 27, 2016

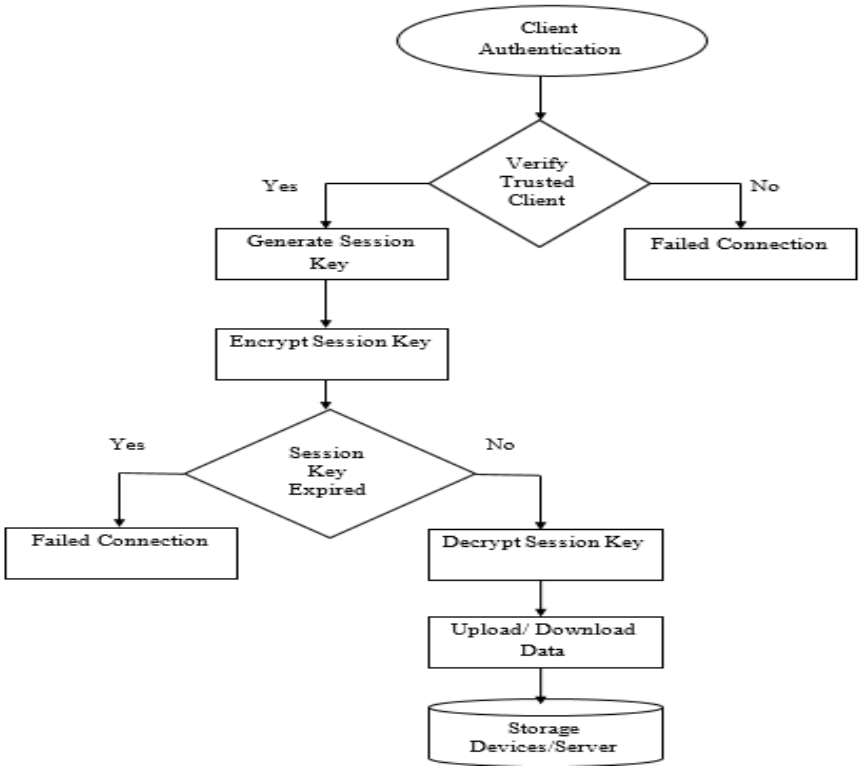

Fig-2 Dataflow model of protocol

Design Goals

In our solutions, we focus on efficiency and scalability with respect to the metadata server. Our main goal is to reduce the workload of the metadata server. Alternatively, the computational and communication overhead for both the client and the storage device should stay convincingly low. Further, we would like an escrow-free solution, that is, the metadata server doesn't learn the session key which is shared between a client and a storage device, unless the server colludes with either one of them. Fig 2 illustrates the Dataflow model of proposed protocols.

6. DESCRIPTION OF PROPOSED PROTOCOL

AE-KEP-I:

Our first protocol can be regarded as a modified version of Kerberos that allows the client to generate its own session keys, i.e. the key objects used to derive a session key is pre-computed by the client for each *v* and forwarded to the equivalent 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. To address key escrow at the same time achieving forward secrecy, we incorporate an Elliptic curve cryptography technique into Kerberos-like AE-KEP-I. Particularly, the client *C* and the storage device *Si* each now chooses a secret value (that is known only to itself) and pre-computes an Elliptic curve cryptography key component. A session key is then generated from both the Elliptic curve cryptography components. Upon expiry of a time period *v*, the secret values and Elliptic curve cryptography key components are permanently erased, such that in the event when either *C* or *Si* is compromised, the attacker or third party will no longer have permission to access the key values required computing past session keys.

AE-KEP-II:

With the use of AE-KEP-I, we achieve only partial forward secrecy (PFS) (with respect to *v*), by trading efficiency over security. This point out that compromise of a long term key can reveal session keys produced within the current *v*. However, past session keys in previous (expired) time periods *v0* (for *v0*<*v*) will not be affected. Our second protocol aims to achieve full forward secrecy (FFS), that is, exposure of a long term key impact only on a current session key (with respect to *t*), but not all the other past session keys. It is also important to prevent key escrow. In a nutshell, we enhance AE-KEP-I with a key update technique based on any efficient one-way function, such as a keyed hash function. In first phase, it requires *C* and each *Si* to share some initial key material in the form of Elliptic curve cryptography. In second phase, 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 doesn't expose information about its pre image, the associated session key is forward secured.

7. SYSTEM DESIGN AND ANALYSIS

A. Parallel Session

The parallel sessions are the parallel secure session between the clients and storage devices which are in the parallel Network File System. This is the most recent internet standard in an efficient and scalable manner. These are similar to the situations where once the adversary co-session key promises the long term secret key. It can learn the subsequence sessions. If a legitimate client and reliable storage device complete the matching sessions then they compute the same session key. Secondly two of our protocol provides forward secrecy. One is particularly forward securing with respect to multiple sessions within a particular time period.

B. Encrypted and Authenticated key exchange

The primary goal of this work is to describe an efficient and secure encrypted and authenticated key exchange protocol that means the specific requirements of pNFS. Two new provably secure authenticates key exchange protocol are the main results of this project. It describes the main goals and gives some simulation of a variety of pNFS Encrypted and Authenticated key exchanged protocols that are consider in the work.

C. Forward secrecy

Forward secrecy is a secure communication protocols in which compromise of long term keys without compromise past session keys. Forward secrecy protects past session keys against future comparison of secret keys or passwords. If forward secrecy is used, encrypted interactions and sessions recorded in the past cannot be retrieved and decrypted should passwords or long-term secret keys be compromised in the future, even if the adversary actively interfered. The protocols supposed to guarantee the security of the previous session keys when the long term secret key of a client or a storage device is some time compromised.

D. Key escrow

Key escrow is a regulation in which the keys needed to encrypt decrypted data are held in escrow so that, under certain situation, an authorized third party or intruder may gain access to those keys. These third parties may include businesses, who may want to accesses the employees' private

communications, or governments, who may wish to be able to view the contents of encrypted communications. To address key escrow while achieving forward secrecy simultaneously, it incorporate an elliptic curve cryptography technique into Kerberos like AE-KEP-I. However it is to be mated that we gain only forward secrecy, by trading efficiency over security.

E. Elliptic curve cryptography

Elliptic curve cryptography (ECC) is a proposal to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC need smaller keys compared to non-ECC cryptography to provide corresponding security. Elliptic curves are relevant for encryption, digital signatures, pseudo-random generators and other tasks. They are also used in several integer factorization algorithms that have applications in cryptography.

CONCLUSION

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

FUTURE ENHANCEMENT

The proposed work mainly focuses on the scalability and forward secrecy problems. It overcomes theses problem while reducing the workload of metadata server. In the future work we can enhance the protocol to avoid the possible security attacks in the parallel network environment which will mainly focus on the key escrow. The performance of the protocol will be increased.

REFERENCES

[1] Hoon Wei Lim and Guomin Yang, “Authenticated Key Exchange Protocols for Parallel Network File Systems” in vol. 27, no. 1, Jan. 2016

[2] M. Abd-El-Malek, W. V. Courtright II, C. Cranor, E. Thereska, M. Wachs, and J. J. Wylie, “Ursa minor: Versatile cluster-based storage,” in Proc. 4th USENIX Conf. File Storage Technol., Dec. 2005,pp. 59–72.

[3] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, M. Theimer, and R. Wattenhofer, “FARSITE: Federated, available, and reliable storage for an incompletely trusted environment,” in Proc. 5th Symp. Oper. Syst. Des.Implementation, Dec. 2002, pp. 1–14.

[4] M. K. Aguilera, M. Ji, M. Lillibridge, T. Mann, and C. A. Thekkath, “Block-level security for network-attached disks,” in Proc. 2nd Int. Conf. File Storage Technol., Mar. 2003, pp. 159–174.

[5] A. W. Leung, E. L. Miller, and S. Jones, “Scalable security for petascale parallel file systems,” in Proc. ACM/IEEE Conf. High Perform. Netw. Comput., Nov. 2007, p. 16.

[6] J. Kubiatowicz, D. Bindel, Y. Chen, and B. Y. Zhao, "OceanStore: An architecture for global scale persistent storage," in Proc. 9th Int. Conf. Arch. Support Program. Language Oper. Syst., Nov. 2000, pp. 190–201.

[7] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in Proc. 19th Int. Conf. Theory Appl. Cryptographic Techn., May 2000, pp. 139–155.

[8] D. Boneh, C. Gentry, and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," in Proc. 25th Annu. Int. Conf. Adv. Cryptol., Aug. 2005, pp. 258–275.

[9] B. Callaghan, B. Pawlowski, and P. Staubach, "NFS version 3 protocol specification," Internet Eng. Task Force (IETF), RFC 1813,Jun. 1995.

[10] F. Hupfeld, T. Cortes, B. Kolbeck, J. Stender, J.Malo, J. Marti, and E. Cesario, "The XtreemFS architecture—A case for object-based file systems in grids," Concurrency Comput.: Prac. Exp., vol. 20, no. 17, pp. 2049–2060, Dec. 2008.
[11] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in Proc. 6th Symp. Oper. Syst. Des. Implementation, Dec. 2004, pp. 137–150.

[12] M. Eisler, "LIPKEY—A low infrastructure public key mechanism using SPKM," Internet Eng. Task Force (IETF), RFC 2847, Jun. 2000.

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

[14] M. Eisler, "RPCSEC_GSS version 2," Internet Eng. Task Force(IETF), RFC 5403, Feb. 2009.

[15] M. Eisler, A. Chiu, and L. Ling, "RPCSEC_GSS protocol specification," Internet Eng. Task Force (IETF), RFC 2203, Sep. 1997.

[16] S. Emery, "Kerberos version 5 generic security service application program interface (GSS-API) channel binding hash agility," Internet Eng. Task Force (IETF), RFC 6542, Mar. 2012.