Authenticated Key Exchange Protocols for pNFS

Amruta Patil[1] Nadeem Pathan[2] Ghanshyam Kolse[3]
[1,2,3]Department of Computer Engineering
[1,2,3]SCSCOE Maharashtra, India

Abstract— We study the problem of key creation for secure many to many communications. The problem is raise by the propagation of large scale distributed file system supporting parallel access to multiple storage devices. Our work focuses on current Internet standards for such file systems, i.e. the parallel Network File System (pNFS), which use of Kerberos to establish parallel session keys between client and storage devices. Our review of the existing Kerberos-based protocol has a number of boundaries: (i) a metadata server facilitating key exchange between clients and storage devices has heavy workload which restricts the scalability of the protocol; (ii) the protocol does not provide forward secrecy; (iii) metadata server establish itself all the session keys that are used between the clients and storage devices, and this inherently leads to the key escrow. In this paper, we propose a variety of authenticated key exchange protocols that are designed to address above issues. We show that our protocols are capable of reducing up to approximately 54% of workload of a metadata server and concurrently supporting forward secrecy and escrow-freeness.

Key words: Parallel sessions, authenticated key exchange, network file systems, forward secrecy, key escrow

I. INTRODUCTION

In the network connected parallel file systems, the data is stored across multiple storage devices or nodes to allow simultaneous accessing by multiple operation of a parallel apps. That mainly used in large number of cluster computing that focuses on high performance and reliable fetch to large datasets. That larger I/O bandwidth is get through simultaneous fetching data from multiple storage devices in large computing clusters, and data loss is prevented through data redundancy by using defect tolerant striping algorithms. Examples of parallel file systems. which are in the production use are the IBM General Parallel Files System. which are usually required for advanced scientific or data intensive. In these environments hundreds or thousands of file system clients distribute data and generate large I/O load on the file system supporting petabytes or terabytes scale storage capacities. Independent of the development of the clusters and clouds and the Map-Reduce model has resulted in file system such as the Hadoop's HDFS. In this task, we study the problem of the secure many to many communications in the large scale NFS which support parallel fetching data from multiple storing devices. That we considering the communication model where there are a crowd of the clients accessing many remotely located and distributed storage devices in parallel. we tries to focus on how to exchange the key materials and establishment of the parallel secure sessions in between storage devices and client in the pNFS, the current Internet standards in efficient and scalable manner.

The development of pNFS is driven by Sun, IBM, EMC, and UMich/CITI, and hence it has many similar features and is compatible with many existing network file systems. Our main aim in this task is to model secure authentication key exchange protocols that meet specific needs of pNFS. Particularly, we attempt to meet the following desirable characteristics, which have not been successfully achieved by current Kerberos-based elucidation.

A. *RELATED WORK:*

1.1 Password-based encrypted key exchange are protocols that are designed to provide pair of users communicating over an unreliable channel with a secure session key even when the secret key or password shared between two users is drawn from a small set of keys. In proposed scheme, two simple passwords based encrypted key exchange protocols based on that of Bellovin and Merritt. While one protocol is more suitable to scenarios in which the password is shared across multiple servers, the other provides better security. Both protocols are as efficient, if not better, as any of the existing encrypted key exchange protocols in the literature, and yet they only require a single random oracle instance. The proof of security for both protocols is in the random oracle model and based on hardness of the computational Diffe-Hellman problem. However, some of the techniques that we use are quite different from the usual ones and make use of new variants of the Diffe-Hellman problem, which are of independent interest. We also provide concrete relations between the new variants and the standard Diffe-Hellman problem. Advantage of this scheme it is possible to find several flavors of key. In this different types of protocols are used like SIGMA, IKE etc. [2]

1.2 In a public network, when a number of clusters connected to each other is increased becomes a potential threat to security applications running on the clusters. To address this problem, a Message Passing Interface (MPI) is developed to preserve security services in an unsecured network. The proposed work focuses on MPI rather than other protocols because MPI is one of the most popular communication protocols on distributed clusters. Here AES algorithm is used for encryption/decryption and interpolation polynomial algorithm is used for key management which is then integrated into Message Passing Interface Chameleon version 2 (MPICH2) with standard MPI interface that becomes ES-MPICH2. This ES-MPICH2 is a new MPI that provides security and authentication for distributed clusters which is unified into cryptographic and mathematical concept. The major desire of ES-MPICH2 is supporting a large variety of computation and communication platforms. The proposed system is based on both cryptographic and mathematical concepts which leads to full of error free message passing interface with enhanced security. [3]

1.3 Password Authenticated Key Exchange (PAKE) is one of the important topics in cryptography. It aims to address a practical security problem: how to establish secure communication between two parties solely based on a shared password without requiring a Public Key Infrastructure (PKI). After more than a decade of extensive research in this field, there have been several PAKE protocols available. The EKE and SPEKE schemes are perhaps the two most notable examples. Both techniques are however patented. In this paper, we review these techniques in detail and summarize various theoretical and practical weaknesses. In addition, we present a new PAKE solution called J-PAKE. Our strategy is to depend on well-established primitives such as the Zero-Knowledge Proof (ZKP). So far, almost all of the past solutions have avoided using ZKP for the concern on efficiency. We demonstrate how to effectively integrate the ZKP into the protocol design and meanwhile achieve good efficiency. Our protocol has comparable computational efficiency to the EKE and SPEKE schemes with clear advantages on security. [4]

1.4 We present a mechanized proof of the password- based protocol One-Encryption Key Exchange (OEKE) using the computationally-sound protocol prover CryptoVerif. OEKE is a non-trivial protocol, and thus mechanizing its proof provides additional confidence that it is correct. This case study was also an opportunity to implement several important extensions of CryptoVerif, useful for proving many other protocols. We have indeed extended CryptoVerif to support the computational Diffie-Hellman assumption. We have also added support for proofs that rely on Shoup’s lemma and additional game transformations. In particular, it is now possible to insert case distinctions manually and to merge cases that no longer need to be distinguished. Eventually, some improvements have been added on the computation of the probability bounds for attacks, providing better reductions. In particular, we improve over the standard computation of probabilities when Shoup’s lemma is used, which allows us to improve the bound given in a previous manual proof of OEKE, and to show that the adversary can test at most one password per session of the protocol. In this paper, we present these extensions, with their application to the proof of OEKE. All steps of the proof, both automatic and manually guided, are verified by CryptoVerif. [5]

1.5 Password-Authenticated Key Exchange (PAKE) studies how to establish secure communication between two remote parties solely based on their shared password, without requiring a Public Key Infrastructure (PKI). Despite extensive research in the past decade, this problem remains unsolved. Patent has been one of the biggest brakes in deploying PAKE solutions in practice. Besides, even for the patented schemes like EKE and SPEKE, their security is only heuristic; researchers have reported some subtle but worrying security issues. In this paper, we propose to tackle this problem using an approach different from all past solutions. Our protocol, Password Authenticated Key Exchange by Juggling (J-PAKE), achieves mutual authentication in two steps: first, two parties send ephemeral public keys to each other; second, they encrypt the shared password by juggling the public keys in a verifiable way. The first use of such a juggling technique was seen in solving the Dining Cryptographers problem in 2006. Here, we apply it to solve the PAKE problem, and show that the protocol is zero-knowledge as it reveals nothing except one-bit information: whether the supplied passwords at two sides are the same. With clear advantages in security, our scheme has comparable efficiency to the EKE and SPEKE protocols. [6]

II. PROPOSED SYSTEM

Particularly, this include a collection of three protocols: (i) the pNFS protocol that transfer file metadata, also called as a layout, between the metadata server and a client; (ii) the storage access protocol that specifies how the client accesses data from the associated storage devices according to the related metadata; and (iii) the control protocol that synchronizes the state between the metadata server and the storage devices.

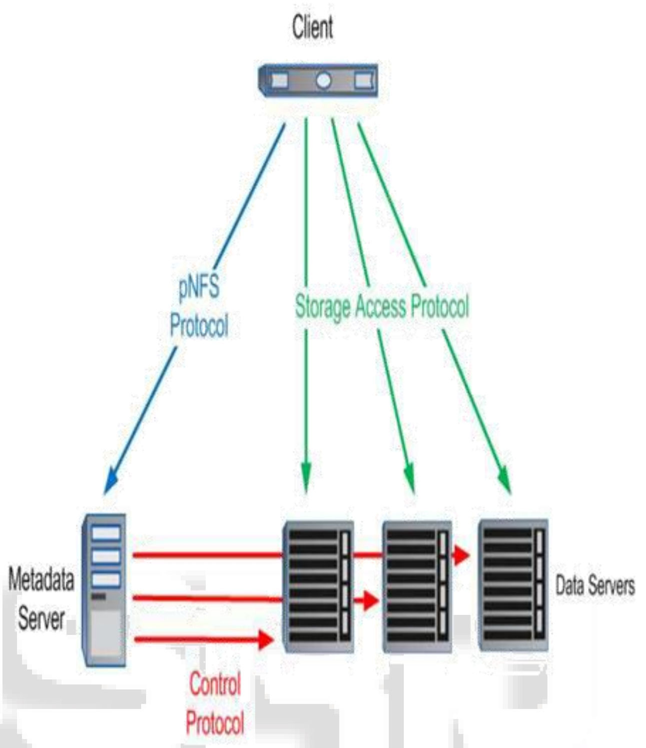

Fig. 1: The Conceptual Model of pNFS

A. *SYSTEM FLOW :*

We have introduced metadata in our work; metadata plays a very important role in running the client operation. Metadata performs the main task of authentication of client. It create One-Time-Password (OTP) to authenticate the client access. Once the client gets confirmed the metadata create session key which enables client to access resources for particular period.

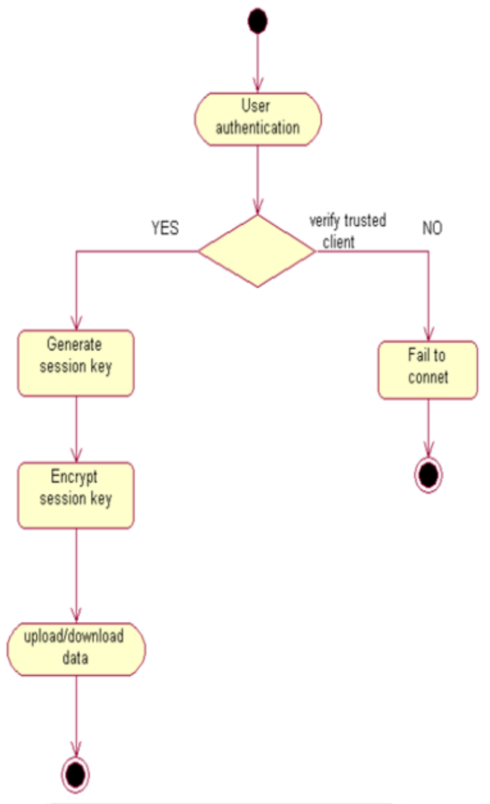

Fig. 2.1: System Flow

B. *ALGORITHM*

Upon receiving an I/O request for a file object from *C*, each *Si* performs the following:

1) check if the layout σi is valid;
2) decrypt the authentication token and recover key KCSi;
3) compute keys skz i = F(KCSi;IDC,IDSi,v,sid,z) for z = 0,1;
4) decrypt the encrypted message, check if IDC matches the identity of C and if t is within the current validity period v;
5) if all previous checks pass, Si replies C with a key confirmation message using key sk0 i.

1) *EXPLANATION OF ALGORITHM*

1) In first step we are checking if available layout is valid or not for further operations and communication.
2) In second step we do the decryption operation on the token which is generated by metadata server for authentication process. By performing decryption, we will recover the key for client set.
3) In this third step we will compute the key for storage set for accessing the data\information within the storage set. We will compute key by checking the key of client set as well as id for users. As per the result we will return access to user or denied to communicate.
4) Fourth step will perform the task of decryption of encrypted message. And it will also check for validation for user access.
5) In this final step if all the above process is successfully validated then it will return key confirmation message to User\client.

III. OUTLINE OF PROTOCOL

– 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.
– 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-I. Particularly, the client C and the storage device Si each now select a secret value (that is known only to itself) and pre-computes a Diffie-Hellman key component. A session key is then generated from both the Diffie-Hellman components.
– 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 to t), but not all the other past session keys.

IV. CONCLUSIONS

We proposed three authenticated key exchange protocols for parallel network file system (pNFS). Our protocols offer the 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, two our protocols provide forward secrecy: one is partially forward secure (with respect to the 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.

ACKNOWLEDGEMENT

We would like to thank Prof. Jadhav H. B. HOD at SCSCOE, for his valuable guidance. We wish to express our sincere gratitude to Prof. Sinare P. D. We sincerely thank Prof. Nita Mhaske to her encouragement in carrying out this project work. We also wish to express our gratitude to the all lecturers of SCSCOE who render their support during the period of project work.

REFERENCES

[1] Hoon Wei Lim Guomin Yang, "Authenticated Key Exchange Protocols for Parallel Network File Systems", IEEE Transactions on Parallel and Distributed Systems 2015.
[2] Michel Abdalla, David Pointcheval., "Simple Password-Based Encrypted Key Exchange Protocols."
[3] R.S.RamPriya, M.A.Maffina., "A Secured and Authenticated Message Passing Interface for Distributed Clusters."
[4] Feng Hao1 and Peter Ryan2., "J-PAKE: Authenticated Key Exchange Without PKI"
[5] Bruno Blanchet., "Automatically Verified Mechanized Proof of One-Encryption Key Exchange"
[6] Feng Hao1 and Peter Ryan2., "Password Authenticated Key Exchange by Juggling"