International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 13
September 2016

An improved authenticated key exchange technique for parallel network file system security

MD Rahimunnisa
PG Scholar,
Department of CSE,
DIET, ANAKAPALLE, Visakhapatnam

A.A. Narasimham
Associate Professor,
Department of CSE,
DIET, ANAKAPALLE, Visakhapatnam

Abstract- *This proposes a variety of authenticated key exchange protocols that are designed to address the issues. We show that our protocols are capable of reducing up to approximately 54% of the workload of the metadata server and concurrently supporting forward secrecy and escrow-freeness. All this requires only a small fraction of increased computation overhead at the client. We proposed three authenticated key exchange protocols for parallel network file system (pNFS).Our work focuses on current Internet standards for such file systems, i.e. the parallel Network File System (pNFS), which makes 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 limitations: (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 the above issues. We show that our protocols are capable of reducing up to approximately 90% of the workload of the metadata server and concurrently supporting forward secrecy and escrow-freeness. All this requires only a small fraction of increased computation overhead at the client. The main technique used here is ECDH which is an anonymous key exchange protocol that allows two parties, each having an elliptic curve public–private key pair, to establish a shared secret over an insecure channel.*

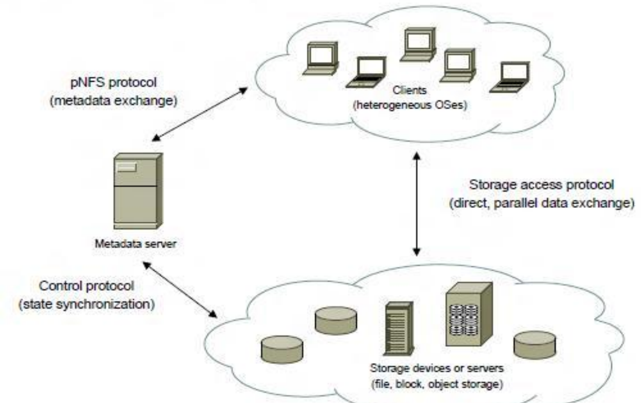
Keywords: Kerberos, Authenticated key exchange, Metadata server, forward secrecy, escrow-freeness Parallel sessions.

1. INTRODUCTION

In parallel file systems, the file data is distributed across various capacity gadgets or hubs to permit simultaneous access by different errands of a parallel application. That is ordinarily utilized as a part of vast scale group registering that spotlights on elite and dependable bring to extensive datasets. That higher I/O transfer speed is accomplished through simultaneous bringing information to various stockpiling gadgets inside of extensive registering bunches, while information misfortune is ensured through information reflecting utilizing deformity tolerant striping calculations. Couple of cases of elite parallel record frameworks that are in the creation use are the IBM General Parallel Files System, which are normally required for cutting edge logical or information escalated applications, for example, advanced movement studios, computational liquid progress, and semiconductor producing the problem to secure many to many communications in large-scale network file systems that support parallel access to multiple storage devices. That is, we consider a communication model where there are a large number 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 exchange key materials and establish parallel secure sessions between the clients and the storage devices in the parallel Network File System (pNFS) the current Internet standard in an efficient and scalable manner.Our most important objective in this vocation is to plan well-organized and protected genuine key swap over procedures that get together unambiguous necessities of pNFS. Predominantly, we challenge to get together the subsequent advantageous possessions, which moreover have not been adequately accomplished or are not attainable by the present Kerberos, we consider the communication model where there are a large number of the clients accessing multiple remote and distributed storage devices in parallel. Particularly, we try to focus on how to exchange the key materials and establishment of the parallel secure sessions between clients and storage devices in the parallel Network File System (pNFS), the current Internet standards in efficient and scalable manner. The development of pNFS is driven by Sun, EMC, IBM, and UMich/CITI, and thus it shares many similar features and is compatible with many existing commercial network file systems. Our main goal in this work is to design efficient and secure authenticated key exchange protocols that meet specific needs of pNFS.

2. Parallel Network File System (PNFS)

NFS was developed by Sun Microsystems and has been designated a file server standard. Its protocol uses the Remote Procedure Call (RPC) method of communication between computers. You can install NFS on Windows 95 and some other operating systems using products like Sun's Solstice Network Client. Using NFS, the user or a system administrator can mount all or a portion of a file system. Network File System (NFS) is currently the sole file system standard supported by the Internet Engineering Task Force (IETF). The NFS protocol is a distributed file system protocol originally developed by Sun Microsystems that allows a user on a client computer, which may be diskless, to access files over networks in a manner similar to how local storage is accessed. The NFS protocol has since then evolved into an open standard defined by the IETF Network Working Group. Among the current key features are file system migration and Replication, file locking, data caching, delegation (from server to client), and crash recovery. 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 stored across 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. Figure 1 illustrates the conceptual model of pNFS.



More specifically, pNFS comprises a collection of three protocols: (i) the pNFS protocol that transfers file metadata, also known as a layout,1 between the metadata server and a client node; (ii) the storage access protocol that specifies how a client accesses data from the associated storage devices according to the corresponding metadata; and (iii) the control protocol that synchronizes state between the metadata server and the storage devices.

2.1 Current Limitations

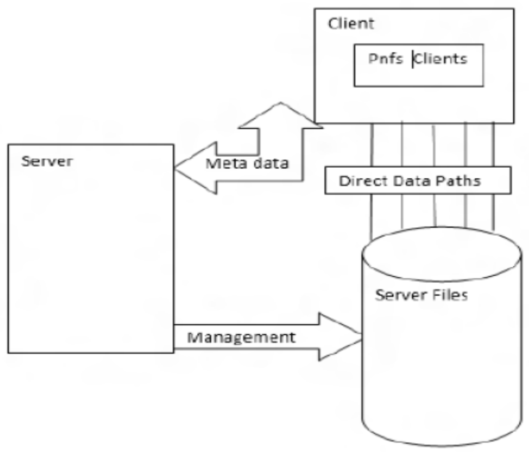
The current design of NFS/pNFS focuses on interoperability, instead of efficiency and scalability, of various mechanisms to provide basic security. Moreover, key establishment between a client and multiple storage devices in pNFS are based on those for NFS, that is, they are not designed specifically for parallel communications. Hence, the metadata server is not only responsible for processing access requests to storage de-vices (by granting valid layouts to authenticated and authorized clients), but also required to generate all the corresponding session keys that the client needs to communicate securely with the storage devices to which it has been granted access. Consequently, the metadata server may become a performance bottleneck for the file system. Moreover, such protocol design leads to key escrow. Hence, in principle, the server can learn all information transmitted between a client and a storage device. This, in turn, makes the server an attractive target for attackers. Another drawback of the current approach is that past session keys can be exposed if a storage device's long-term key shared with the metadata server is compromised. We believe that this is a realistic threat since a large-scale file system may have thousands of geographically distributed storage devices. It may not be feasible to provide strong physical security and network protection for all the storage devices

3. Related Work

The large number key scheme has been proposed. In existing system, we used in large number clients and one server. In proposed system used also large number clients and one type of meta server. The previous system defined a large scale distributed system. Large scale distributed means more number of including them. The existing system both meta server and storage devices are trusted entities, no implicit trust on the clients. The storage devices are accessed only in company and college area. The key verification technique is applied accessing files in many communication processes. The whole process is a client should enter the details into the server, server should accept the client, server accept the all clients because only occur in authorized clients. Passwords are a standout amongst the most well-known reasons for framework crashes, in light of the fact that the low entropy of passwords makes frameworks helpless against beast power speculating assaults. Because of new innovation passwords can be hacked effectively. Robotized Turing Tests keep on being a viable, simple to send way to deal with distinguish mechanized malevolent login endeavors with sensible expense of burden to clients. Subsequently in this proposed plan the insufficiency of existing and proposed login conventions intended to address substantial scale online lexicon assaults e.g. from a botnet of a huge number of hubs. In this plan proposed a basic plan that fortifies watchword based verification conventions and forestalls online word reference assaults and also numerous to-numerous assault regular to 3-pass SPAKA conventions The client should share the information to the other clients. At the time server should generate the one time password key. Next step, the client uploads the file to meta server. Then other clients download the files, the server should generate key and send into a client's mail, the clients can pass the key in server, then server lead file into the clients. So, in this process no occurrence of unauthorized clients, then it also generates authentication. The related work is more dependent on existing system. It is called as proposed system. Key swapping are conventions that are intended to give pair of clients conveying over an inconsistent channel with a safe session key notwithstanding when the mystery key or secret word shared between two clients is drawn from a little arrangement of keys. In proposed plan, two basic passwords based encoded key swapping conventions in light of that of Bellovin and Merritt. While one convention is more suitable to situations in which the secret key is shared over various servers, alternate gives better security. Both conventions are as productive, if worse, as any of the current scrambled key swapping conventions in the writing, but they just require a solitary arbitrary prophet case. The evidence of security for both conventions is in the irregular prophet show and in view of hardness of the computational Diffe-Hellman issue. Be that as it may, a percentage of the methods that we utilize are entirely not quite the same as the typical ones and make utilization of new variations of the Diffe-Hellman issue, which are of free hobby. We likewise give solid relations between the new variations and the standard Diffe-Hellman issue. Favorable position of this plan is, it is conceivable to discover a few kinds of key.

4. IMPLEMENTATION

The vital of this proposed work is to provide security to Meta server data in parallel network file system. The key verification technique is applied accessing files in many-many communication process. This paper is unauthenticated as key generation scheme are high security by using authentication process. The proposed system used as the Fault Tolerant striping protocol. It is defined as If look at the words fault and tolerance, can define the fault as a malfunction from normal behavior and tolerance as the capacity for enduring or putting up with something. The fault tolerance refers to a system's ability to deal with malfunctions. This is mainly used in Internet environment. This risk widely extent in the internet. This risk occurs in security attack, to solve in this problem in security mechanism. The server generates key, the key range in 32bit.The Security mechanism is used to detect, prevent or recover from a security attacks. The mechanisms are divided into those that implemented in a exact protocol sheet and those that are not exact to any particular protocol sheet or security service.



The metadata is intermediated between client and server. The clients maintain a parallel network file. Meta Server manages the server files. Clients access the server files through direct data paths.

4.1 Client authentication

In this module each new client must be created their own profile to access the meta server. In registration process authentication information is provided to each user. Using this authentication only they log on to the application and access it. The information provided to each User must be unique and confidentially maintained.

4.2 Server verification

In this module, a server verifies client information and accepts it. Then only the clients access the application. Thus every time when the client access the application a new one time password key is generated and send to their mail id. Using this key only the client access the server and go further process. This authentication process is verified each time. Server monitors all clients file Details and data sharing information also.

4.3 Data sharing

In this module an authorized client upload their data to the Meta server. This information is accessed by many other clients in secure manner. A client uploads file information such as file type, File name to the server. Clients send a message notification to other clients about their data Sharing details.

4.4. Key generation

In this module clients access the meta server data in secure manner. First client searches data from the server based on the category. All file information is shown such as file name, file type and client details who upload data to the server. The client will access files using the key only. When the client requests a key for accessing the file, a unique is generated and sends it to their mail id.

4.5 Data access

In this module a client access the files from the server using a key. Clients get a key from their Mail id each time accessing the file. The key is verified, then only the client able to download the File securely. This key generation and verification mechanism is applied for all files in each time Access. The server accesses the data to the client, by using the server files. The several clients can access the data in parallel network. The existing system will be used as the key sharing midpoint, but in the proposed system the key sharing midpoint is an important one.

5 CONCLUSION

We proposed three advanced authenticated key exchange protocols for cloud and parallel network file system (pNFS). Our protocols offer three appealing advantages over the existing. 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. It should occur in three advantages, i) it also provide the better scalability. ii.)The protocols produce forward secrecy. iii) 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 complicity among them. The week password is used to develop a key that encrypts and transmitted between client and key sharing midpoint.

References

[1] M. Abd-El-Malek, W.V. Courtright II, C. Cranor, G.R. Ganger, J. Hendricks, A.J. Klosterman, M.P. Mesnier, M. Prasad, B. Salmon, R.R. Sambasivan, 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 59–72. USENIX Association, Dec 2005.

[2] C. Adams. The simple public-key GSS-API mechanism (SPKM). The Internet Engineering Task Force (IETF), RFC 2025, Oct 1996.

[3] A. Adya, W.J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J.R. Douceur, J. Howell, J.R. Lorch, M. Theimer, and R. Wattenhofer. FARSITE: Federated, available, and reliable storage for an incompletely trusted environment. In Proceedings of the 5th Symposium on Operating System Design and Implementation (OSDI). USENIX Association, Dec 2002.

[4] M.K. Aguilera, M. Ji, M. Lillibridge, J. MacCormick, E. Oertli, D.G. Andersen, M. Burrows, T. Mann, and C.A. Thekkath. Blocklevel security for network-attached disks. In Proceedings of the 2nd International Conference on File and Storage Technologies (FAST).USENIX Association, Mar 2003.

[5] 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.

[6] Amazon simple storage service (Amazon S3). http://aws.amazon.com/ s3/.

[7] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In Advances in Cryptology – Proceedings of EUROCRYPT, pages 139–155. Springer LNCS 1807, May 2000.

[8] 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 2005.

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

[10] 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.

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

[12] 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

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

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

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

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

[18] 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.

[19] 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.
[20] Financial Services Grid Initiative. http://www.fsgrid.com/.

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

[22] 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.

[24] 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.

[25] 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.

AUTHORS

MD Rahimunnisa
PG Scholar,
Department of CSE,
DIET, ANAKAPALLE,
Visakhapatnam

A.A.Narasimham
Associate Professor,
Department of CSE,
DIET, ANAKAPALLE,
Visakhapatnam