2011 International Conference on Information and Electronics Engineering
IPCSIT vol.6 (2011) © (2011) IACSIT Press, Singapore

Hierarchal Hash Based Secure Network File System for Wireless Networks

Shehzad Ashraf Ch, Waqas Nasar, Anwar Ghani , Muhammad Sher
International Islamic University Islamabad, Pakistan
{Shahzad, waqas.nasar, anwar.mscs429, m.sher}@iiu.edu.pk

Abstract. Running of ordinary file systems on wireless networks can result in short battery life and bandwidth dilemma for large file sharing, which results in high latency and unbearable delays which user do not consider while working on wireless networks, also the security is second main issue in wireless networks as any false node can inject his packet or private information can be divulged to some one who is not intended, so there is paramount need of an efficient and secure file system for wireless networks. The papers describes the need of secure and bandwidth efficient file system for wireless networks and also proposes a file system which is secure and bandwidth efficient and can become a part and parcel of wireless networks

Keywords: Network file system, Efficient NFS, File system security, wireless file system

1. Introduction

The responsibility of a file system is to store, manage, copy, move, modify the files, which is a basic unit in some file system, this unit contains metadata and contents or information stored in it, all above mentioned operation can be performed on both file metadata and file contents. The network file system is responsible for all above mentioned tasks of a file on a network as well as performing three more tasks related to network which are remote procedure call, External data representation and operation of layer seven of OSI reference model.

Remote procedure call is performed at session layer its purpose is to provide access to different procedures residing on remote machine, The representation of data of external devices is to uniform distribution of storage capacities of devices present on a network and to access all these storage devices smoothly by all the nodes of a network. There are two part and parcels of Network File system Which are Location transparency and Location independence, The purpose of location independence is to name the files such that the name does not expose the storage location of the file and location independence means if the physical location of a file is changed then its name should not be changed. The purpose of these two characteristics of network file system makes the file user feel comfortable while dealing with files, that is he do not worry where the file is going to store or if the location of file is changed then how he will access the file, the file user will perform all the operations as the file is residing on his own machine. The main problem being faced by user / designers of a network file system is the cache consistency problem which to keep the cache copies consistent with the master file

2. Literature Review

The consideration of high bandwidth consumption over slow network is an important feature for performance and efficiency, the issue got less attention as for as user perspective is considered [BP]. In [1] the author proposed WAN acceleration model for developing world, the idea is based on reducing network traffic by avoiding data which is already present in receiver cache. The data is broken into variable size chunks; each chunk will have a corresponding integer number, these integer numbers will be used for decision of sending the chunk onto WAN. Author proposed Multi resolution chunking (MRC) which use robin finger prints for chunk boundaries, MRC creates a hierarchal aligned chunk boundaries, all of the smallest boundaries are detected, and then larger chunks are generated by matching different numbers of bits of the same boundary detection constraint, here the largest chunk is the root, and all smaller chunks share boundaries with some of their leaf chunks. Performing this process using one fingerprinting pass not only produces a cleaner chunk alignment, but also requires less CPU. [2] Described a scalable de-duplication technique for non-traditional backup workloads that are made up of individual files with no locality among consecutive files in a given window of time. [2] Also claimed that Due to lack of locality, existing techniques perform poorly on these workloads. Extreme Binning exploits file similarity instead of locality, and makes only one disk access for chunk lookup per file, which gives reasonable throughput. In scheme of [2] Each file is allocated using a stateless routing algorithm to only one node, allowing for maximum parallelization, and each backup node is autonomous with no dependency across nodes,

Article [3] is about to make Grid Security Infrastructure (GSI), provides new capabilities to be used on top of different Grid middle wares. The solution is specifically implemented on gLite and accomplishes the access to data storage Grid resources in a uniform and transparent way, the proposed approach of [3] is based on XML data encapsulation to store encrypted data and metadata as well as data owners, X509v3 certificates, AES keys, data size, and file format.

[4] Proposed a file system support for low-bandwidth channels typically the wireless and mobile ad-hoc networks where devices are mobile, typically the problem arises for wireless networks where browsing, downloading and exchanging files can halt the network traffic and a situation of deadlock occurs, the problem is increased twofold when mobile device users wants to share large files or want to collaborate each other. Two approaches were adopted before the proposed solution 1) Fetch the file metadata only and 2) Fetch both metadata and file contents Metadata just gives information about file size, type etc which in some cases is not appropriate (e.g. video files etc), Option 2 is impractical because of resource constraints.

In proposed remote file system [4] the metadata of the file is modified and thumbnail information is added, when a user want to access a file remotely, first of all only metadata of the file will be sent to him, as metadata also contains thumbnail so after seeing thumbnail user can decide whether to open the file or not, so author claimed to reduce significantly bandwidth consumption.

In [5] a new technique for protecting sensitive contents in view only file system is proposed. The authors intended to meet the challenges such as spyware, removable media, and mobile devices which make the sensitive contents disclosed. VOFS relays on trusted computing primitives and virtual machine technology to provide a much greater level of security in the current system. VOFS clients disable non essential device output before allowing the user to view the data to his or her non-secure VM. When the user has finished his work, VOFS reset the machine to previous state and resume normal device activity. It is difficult to prevent data linkage from authorize user, but VOFS do it to some extent using VMM (virtual machine monitor) technology. The VMM has complete control over the computer. In the client architecture above the VMM are the guest virtual machine VM, and is referred to as primary guest VM. This machine has not sufficient access to the resources. Another guest virtual machine i-e domain 0, has complete administrative access to all the resources. A third guest say SVFS VM is responsible for downloading sensitive contents, storing it, and telling the domain 0, when to save or restore primary guest state and when to enable or disable device output. The author claimed that the VOFS clients also get benefit of trusted platform module (TMP) technology. The clients also take the advantage by using an integrity measurement architecture which uses trusted boot to enable remote verification of trusted components by content providers. If a user wishes to view a view only file in VOFS, then the primary VM will make a request to SVFS VM. It will contact the content provider to start an authorization session.

In [6] author claimed that the conventional method (reference monitor) used for authentication was purely centralized, now the world is moving towards distributed environment for this requirement to be fulfilled centralized approach should not be change. In conventional reference monitor method there was the problems of scalability and resource protection. In traditional reference monitor method ACL (access control list) was maintained, every user must be entered in the ACL to have access to the resources. ACL was not scalable well due to ACL, so there should be a way to modify the ACL approach to give access to the new user. Another drawback of ACL was the need of the full time monitoring the requests for resources. Protection and scalability were also the issues to be discussed in conventional approach. The author also proposed a method of cryptographic Access Control. This method eliminates the centralized approach and handles the resource protection as well as the scalability issues. In proposed method encryption is applied at the client and the encrypted data is stored at the server without decrypting the file. The creator of any file is the generator of the keys for encryption and decryption and he is the only who can modify the file. The server saves the file along with keys and the server also maintains the binary tree at server side so fast access can be done in case of searching of the keys. The benefit of saving the key along the file is if any trouble with the binary tree happened then the key is not compromised. Due to the distributed in the nature the proposed method provides scalability, controlled access to the resources, gets rid from constant monitoring because of the encryption the only user can decrypt the file who owned the decryption key (means only the trusted user). The author claimed that authentication, integrity are achieved through digital signature and hash function.

3. Problem Statement

Recent advances in wireless networks led to many promising applications, As these applications are increasing rapidly so the secured communication also becomes the most important issue in wireless networks along with low bandwidth. The consideration of high bandwidth consumption over slow wireless network is an important feature for performance and efficiency. The issue got less attention as far as user perspective is considered. It became more important in interactive session on slow networks, where performance is key issue. Over slower networks interactive programs freeze, like during file I/O, batch commands the execution time increases manifold. In case of File I/O either user has to keep local copies of each or they can use remote login and edit the same file where it is placed. Remote login becomes frustrating for long latency networks. Wireless networks require more security and having low bandwidth than traditional wired networks, so a file system is required which ensures reliable and low bandwidth network operations.

4. Proposed Solution

Architecture of proposed solution is as follows; first the hash code of the file will be generated. Then the file will be broken into two chunk of equal size and their hash codes will be generated. These chunks will further divided be into four chunks and hash code of each chunk will be calculated and so on till an appropriate level k where k=log2n as in Figure 1.

Hash code calculated for each chunk will form a hierarchal hash tree of chunks.

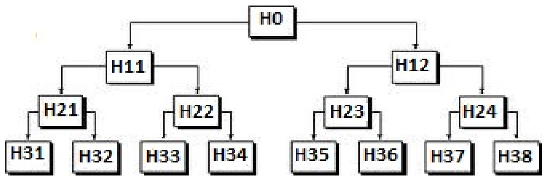

Fig. 1: Hash code Matching

The file will be encrypted each time before write back

Encrypted Hash code of the whole file will be sent to client. Client will decrypt it and match with the Hash code of file already present in his memory. If result is same, this implies that Server and Client having same version of the file, if result is different than client will ask next level hash code of hash tree present in server memory, at each level hash codes will be compared and the chunk having different data will be identified, after identification of chunk whose data has been changed, client will ask server to send the specific chunk, Encrypted data will be transmitted on communication channel by server and client, If client & server data chunks will produce same code, they will not transmit data.

Figure 2 describes the working process and methodology. If a client brought a file from the server in its local cache and don't do any work on it and start some other work. After some time it need to do some work on that file, but it is a possibility that may be during that time some other client had taken that file and updated that file or may not. To ensure consistency and freshness, it needs to consult the server to check whether the file is it in the same state when it took or it has been updated. Then it will consult the server and will send it File name and File handle. In return the server too will send file Name, File handle and the generated tree to the client then the client will compare the both trees in hierarchal way and after that process it will send File Name, File Handle and the Number of the Hash Number mismatched and in return to that the server will send the updated data only instead of the whole file.

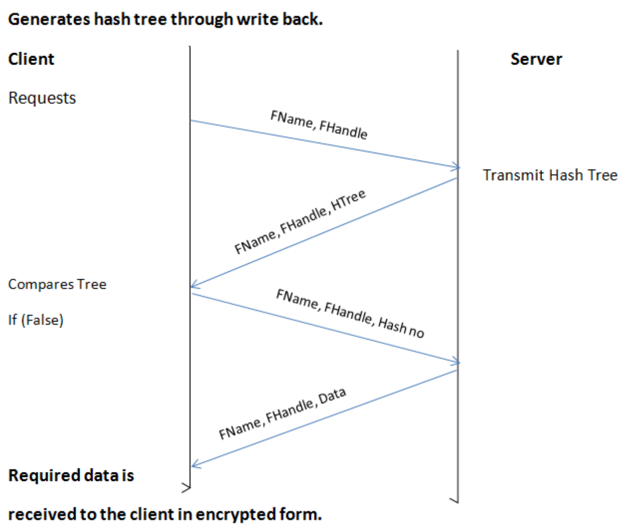

Fig. 2: Flow Chart of Proposed Solution

5. Conclusion

The proposed architecture can save bandwidth by taking advantage of commonality between files, ensures integrity and freshness of a remote file, Operations such as editing documents and compiling software, proposed architecture can consume over an order of magnitude less bandwidth than traditional file systems and performs efficient use of communication link. It also makes transparent remote file access a viable and less frustrating alternative to running interactive programs on remote machines. To ensure minimum running time for Hash matching we used hierarchal hash tree

6. Acknowledgements

Mr. Shehzad Ashraf Chaudhry is being supported by a travel grant from HIGHER Education Commission, Pakistan.

7. References

[1] S.Ihm, K. Park, V.S.Pai, “Wide Area Network Acceleration for the developing World”, 2010

[2] Deepvali Bhagwat, Kave Eshghi, “Extreme Binning: Scalable Parallel Deduplication for chunk Based File Backup, 2009

[3] F. Tusa, M. Villari and A. Puliafito “Design and Implementation of an XML-based Grid File Storage System with Security Features”, 2009

[4] Deepavali Bhagwat, Kave Eshghi, Darrell D.E. Long “Extreme Binning: Scalable, Parallel Deduplication for Chunk-based File Backup” HPL-2009-10R2

[5] Jonathan Ledlie, “File system for low-Bandwidth Thumbnails”, Nokia research center Cambridge, US, May 6, 2008

[6] Securing Sensitive Content in a View-Only File System Kevin Borders, Xin Zhao, Atul Prakash University of Michigan Oct 30, 2006 ACM

[7] Securing Sensitive Content in a View-Only File System Kevin Borders, Xin Zhao, Atul Prakash University of Michigan , 2006

[8] Cryptographic Access Control in a Distributed File System, Anthony Harrington, Christian Jensen, ACM 2003

[9] William J. Bolosky John R. Douceur David Ely Marvin Theimer “Feasibility of a Server less Distributed File System Deployed on an Existing Set of Desktop PCs” 2003

[10] David Mazi`eres. A toolkit for user-level file systems. In Proceedings of the 2001 USENIX Technical Conference,

Boston, MA, June 2001.

[11] Haifeng Yu and Amin Vahdat. Design and evaluation of a continuous consistency model for replicated services. In Proceedings of the 4rd Symposium on Operating Systems Design and Implementation, San Diego, CA, 2000.

[12] Bj¨orn Gr¨onvall, AssarWesterlund, and Stephen Pink. The design of a multicast-based distributed file system. In Proceedings of the Third Symposium on Operating System Design and Implementation, pages 251–264, New Orleans, LA, February 1999.

[13] David Bindel, Yan Chen, Patrick Eaton, Dennis Geels, Ramakrishna Gummadi, Sean Rhea, Hakim Weatherspoon, Westley Weimer, Westley Weimer, Christopher Wells, Ben Zhao, and John Kubiatowicz. Oceanstore: An exteremely wide-area storage system. In Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems, pages 190–201, Boston, MA, November 2000.

[14] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The Click modular router. ACMTransactions on Computer Systems, 18(4):263–297, November 2000.

[15] Andrei Broder. On the resemblance and containment of documents. Compression and Complexity of Sequences, pages 21–29, 1997.

[16] FIPS 180-1. Secure Hash Standard. U.S. Department of Commerce/N.I.S.T., National Technical Information Service, Springfield, VA, April 1995.

[17] Yui-Wah Lee, Kwong-Sak Leung, and M. Satyanarayanan. Operation-based update propagation in a mobile file system. In Proceedings of the 1999 USENIX Technical Conference, Monterey, CA, June 1999.