AN EFFICIENT SECURITY KEY MONITORING SYSTEM FOR ENSURING DATA INTEGRITY IN CLOUD

Dr.R.Reka[1],Dr.M.Nithya[2]
[1]Professor, Department of Information Technology, Panimalar Institute of Technology, Chennai.
[2]Professor & Head, Department of Computer Science & Engineering,
V.M.K.V. Engineering College, Salem

ABSTRACT
Cloud Storage System has a collection of storage servers provides long-standing storage services over the internet. Data privacy becomes a major concern in cloud storage system because user stores his data in third party cloud system. Encryption schemes available for data privacy but it limit the number of functions done in storage system. Building a secure storage system that supports multiple functions is tough when the storage system is distributed and has no central authority. A new idea is proposed proxy re-encryption scheme for decentralizes erasure code for defending the distributed system. The distributed storage system not only supports secure and robust data storage and recovery, but client onward his information in the storage servers to another user without retrieving the data back. The important technical part is that the proxy re-encryption scheme supports encoding operations over encrypted data as well as forwarding operations over encoded and encrypted data. Our schemes fully integrate encoding, encrypting, and onward.

1. INTRODUCTION

Cloud Computing moves the application software and databases to the centralized large data centers, where the management of the data and services may not be fully trust worthy. Building a secure storage system that supports multiple functions is tough when the storage system is distributed and has no central authority. A new idea is proposed proxy re-encryption scheme for decentralizes erasure code for defending the distributed system.

2. LITERATURE SURVEY

2.1 Many-Task Computing in Cloud

The author has defined a new paradigm Many-task computing (MTC), which aims to bridge the gap between two computing paradigms, HTC and HPC. MTC to bridge the gap between two computing paradigms, high throughput computing and high performance computing. MTC differs from high throughput computing in the emphasis of using large number of computing resources over short periods of time to accomplish many computational tasks, where primary metrics are measured in seconds, as opposed to operations per month. MTC notes high-performance computations comprising multiple distinct activities, coupled via file system operations. Tasks may be small or large, uniprocessor or multiprocessor, compute intensive or data-intensive. The set of tasks may be static or dynamic, homogeneous or heterogeneous, loosely coupled or tightly coupled. The aggregate number of tasks, quantity of computing, and volumes of data may be extremely large.

MTC applications are typically loosely coupled that are communication-intensive but not naturally expressed using standard message passing interface commonly found in high performance computing, drawing attention to the many computations that are heterogeneous but not "happily" parallel. The author believes that today's existing HPC systems are a viable platform to host MTC applications. The author also believe MTC is a broader definition than HTC, allowing for finer grained tasks, independent tasks as well as ones with dependencies, and allowing tightly coupled applications and loosely coupled applications to coexist on the same system.

2.2 Secured services in Cloud

Different methods have been proposed to dynamically provide scientific applications with execution environments that hide the complexity of distributed infrastructures. Recently virtualization has emerged as a promising technology to provide such environments. In this work the author has presented an elastic architecture for clusters that allows a flexible management of these computing platforms by: Supporting the execution of heterogeneous application domains; dynamically partitioning the cluster capacity, adapting it to variable demands; and efficiently isolating the cluster workloads. Moreover, this architecture is able to transparently grow the cluster's capacity using an external cloud provider. In this work the author present a generic cluster architecture that extends the classical benefits of virtual machines to the cluster level, so providing cluster consolidation, cluster partitioning and support for heterogeneous environments. Additionally the capacity of the virtual clusters can be supplemented with resources from a commercial cloud provider.

The author has evaluated this architecture in the execution of HTC workloads additionally, and based in this fact, the author has developed a performance model for elastic clusters using cloud resources. This simple model considers the base performance of the cluster and the computational characteristics of the remote cloud. Based on this model it is straightforward to plan the capacity of the cluster to, for example, meet a deadline to complete a given workload. The author envision the use of these kinds of models by additional components to dynamically change the cluster capacity according to a given budget, performance policy or in conjunction with a run and queue wait time prediction service. Finally, the architecture presented in this work is compatible with the use of physical resources. These resources can be devoted to high demanding workloads, such as high performance computing MPI jobs, that will perform badly in virtualized or cloud environments.

2.3 Data Integrity in Cloud Grid

The author describe a system for creating personal clusters in user-space to support the submission and management of thousands of compute-intensive serial jobs to the network-connected compute resources on the NSF TeraGrid. The system implements a robust infrastructure that submits and manages job proxies across a distributed computing environment. These job proxies contribute resources to personal clusters created dynamically for a user on-demand. The system adapts to the prevailing job load conditions at the distributed sites by migrating job proxies to sites expected to provide resources more quickly. The version of the system described in this paper allows users to build large personal Condor and Sun Grid Engine clusters on the TeraGrid. Users can then submit, monitor and control their scientific jobs with a single uniform interface, using the feature-rich functionality found in these job management environments. The Condor version of the system is currently in production deployment on the NSF TeraGrid. Up to 100,000 jobs have been submitted through the system to date, mainly through the Caltech CMS group and the NVO project, enabling approximately 900 teraflops of scientific computation. The system enables users to benefit from a learn-once-run-anywhere submission environment, and resource providers' benefit by not needing to accommodate disruptive changes in their local cluster configuration.

2.4 Dynamic Key Generator in Cloud Clusters

This paper presents new mechanisms for dynamic resource management in a cluster manager called Cluster-on-Demand (COD). COD allocates servers from common pool to multiple virtual clusters (vclusters), with independently configured software environments, name spaces, user access controls, and network storage volumes. The author present experiments using the popular Sun GridEngine batch scheduler to demonstrate that dynamic virtual clusters are an enabling abstraction for advanced resource management in computing utilities and grids. In particular, they support dynamic, policy-based cluster sharing between local users and hosted grid services, resource reservation and adaptive provisioning, scavenging of idle resources, and dynamic instantiation of grid services. These goals are achieved in a direct and general way through a new set of fundamental cluster management functions, with minimal impact on the grid middleware itself.

3. DESIGN OF EXISTING CLOUD STORAGE SYSTEM

The design of existing is to operate a cloud storage system for robustness, privacy and functionality. A cloud storage system is measured as a large-scale distributed storage system that consists of many self-determining storage servers. Data robustness is a major requirement for storage systems. There have been many proposals of storing data over storage servers. The encoding process for a message can be split into n parallel responsibilities of generating code word cipher. A decentralized erasure code is suitable to use in a distributed storage system. After the message cipher are sent to storage servers, each storage server independently computes a code word sign for the received message cipher and stores it. This finishes the encoding and storing process. Disadvantage of Existing system as follows:

- Encoding is not involved in existing system.
- Try and get the success in hacking of data by the third party transfers.
- Transmission is based on third party server to server to client.
- Loss of time in intermediate transmission.

4. DESIGN OF PROPOSED CLOUD STORAGE SYSTEM

Our design concentrates on the problem to forward a data to another user by storage servers directly under the authority of the data owner. We consider the system representation that consists of distributed storage servers and key servers. Since storing cryptographic keys in a single device is uncertain, a user distributes his cryptographic key to key servers that shall perform cryptographic functions on behalf of the user. These key servers are highly secluded by security mechanisms. To well fit the distributed structure of systems. We require that servers to independently perform all operations. With this deliberation, we suggest a new threshold proxy re-encryption scheme and combine it with a secure decentralized code to form a secure distributed storage system. The tight combination of encoding, encryption, and forwarding makes the storage system efficiently meet the requirements of data robustness, data confidentiality, and data forwarding. Our system meets the requirements of storage server's independently performing encoding and re-encryption, and key servers independently perform partial decryption. This makes our system to be more effective.

- A secure cloud storage system implies that an unauthorized user or server cannot get the content of stored messages.
- A storage server cannot generate re-encryption keys by himself.
- Each storage server independently performs encoding and re-encryption and each key server independently perform partial decryption.

CONCLUSION AND FUTURE WORK

We integrate a newly proposed threshold proxy re-encryption scheme and erasure codes over exponents. The threshold proxy re-encryption scheme supports encoding, forwarding, and partial decryption operations in a distributed way. To decrypt a message of k blocks that are encrypted and encoded to n codeword symbols, each key server only has to partially decrypt two codeword symbols in our system. By using the threshold proxy re-encryption scheme, we present a secure cloud storage system that provides secure data storage and secure data forwarding functionality in a decentralized structure. Moreover, each storage server independently performs encoding and re-encryption and each key server independently perform partial decryption. Our storage system and some newly proposed content addressable file systems and storage system are highly compatible. Our storage servers act as storage nodes in a content addressable storage system for storing content addressable blocks. Our key servers act as access nodes for providing a front-end layer such as a traditional file system interface. Further study on detailed cooperation is required.

FUTURE WORK

Our storage system and some newly proposed content addressable file systems and storage system are highly compatible. Our storage servers act as storage nodes in a content addressable storage system for storing content addressable blocks. Our key servers act as access nodes for providing a front-end layer such as a traditional file system interface. Further study on detailed cooperation is required.

REFERENCES

[1] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R.Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, "Oceanstore: An Architecture for Global-Scale Persistent Storage," Proc. Ninth Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS), pp. 190-201, 2000.

[2]P. Druschel and A. Rowstron, "PAST: A Large-Scale, Persistent Peer-to-Peer Storage Utility," Proc. Eighth Workshop Hot Topics in Operating System (HotOS VIII), pp. 75-80, 2001.

[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," Proc. Fifth Symp. Operating System Design and Implementation (OSDI), pp. 1-14, 2002.

[4]A. Haeberlen, A. Mislove, and P. Druschel, "Glacier: Highly Durable, Decentralized Storage Despite Massive Correlated Failures," Proc. Second Symp. Networked Systems Design and Implementation (NSDI), pp. 143-158, 2005.

[5]Z. Wilcox-O'Hearn and B. Warner, "Tahoe: The Least-Authority Filesystem," Proc. Fourth ACM Int'l Workshop Storage Security and Survivability (StorageSS), pp. 21-26, 2008.

[6]H.-Y. Lin and W.-G. Tzeng, "A Secure Decentralized Erasure Code for Distributed Network Storage," IEEE Trans. Parallel and Distributed Systems, vol. 21, no. 11, pp. 1586-1594, Nov. 2010.

[7]D.R. Brownbridge, L.F. Marshall, and B. Randell, "The Newcastle Connection or Unixes of the World Unite!," Software Practice and Experience, vol. 12, no. 12, pp. 1147-1162, 1982.

[8]R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, and B. Lyon, "Design and Implementation of the Sun Network Filesystem," Proc. USENIX Assoc. Conf., 1985.

[9]M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable Secure File Sharing on Untrusted Storage," Proc. Second USENIX Conf. File and Storage Technologies (FAST), pp. 29-42, 2003.

[10] S.C. Rhea, P.R. Eaton, D. Geels, H. Weatherspoon, B.Y. Zhao, and J. Kubiatowicz, "Pond: The Oceanstore Prototype," Proc. Second USENIX Conf. File and Storage Technologies (FAST), pp. 1-14, 2003.

[11]R. Bhagwan, K. Tati, Y.-C. Cheng, S. Savage, and G.M. Voelker, "Total Recall: System Support for Automated AvailabilityManagement," Proc. First Symp. Networked Systems Design andImplementation (NSDI), pp. 337-350, 2004.

[12]A.G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Ubiquitous Access to Distributed Data in Large-Scale Sensor Networks through Decentralized Erasure Codes," Proc. Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN), pp. 111-117, 2005.

[13]A.G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Decentralized Erasure Codes for Distributed Networked Storage," IEEE Trans. Information Theory, vol. 52, no. 6 pp. 2809-2816, June 2006.

[14]M. Mambo and E. Okamoto, "Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts," IEICE Trans. Fundamentals of Electronics, Comm. and Computer Sciences, vol. E80-A, no. 1, pp. 54-63, 1997.

[15]M. Blaze, G. Bleumer, and M. Strauss, "Divertible Protocols and Atomic Proxy Cryptography," Proc. Int'l Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT), pp. 127-144, 1998.

[16]G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage," ACM Trans. Information and System Security, vol. 9, no. 1, pp. 1-30, 2006.

[17]Q. Tang, "Type-Based Proxy Re-Encryption and Its Construction," Proc. Ninth Int'l Conf.

Cryptology in India: Progress in Cryptology (INDOCRYPT), pp. 130-144, 2008.

[18]G. Ateniese, K. Benson, and S. Hohenberger, “Key-Private Proxy Re-Encryption,” Proc. Topics in Cryptology (CT-RSA), pp. 279-294,2009.

[19]J. Shao and Z. Cao, “CCA-Secure Proxy Re-Encryption without Pairings,” Proc. 12th Int’l Conf. Practice and Theory in Public Key Cryptography (PKC), pp. 357-376, 2009.

[20]G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable Data Possession at Untrusted Stores,” Proc. 14th ACM Conf. Computer and Comm. Security (CCS), pp. 598-609, 2007.

[21]G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, “Scalable and Efficient Provable Data Possession,” Proc. Fourth Int’l Conf. Security and Privacy in Comm. Netowrks (SecureComm), pp. 1-10, 2008.

[22]H. Shacham and B. Waters, “Compact Proofs of Retrievability,” Proc. 14th Int’l Conf. Theory and Application of Cryptology and Information Security (ASIACRYPT), pp. 90-107, 2008.

[23]Hsiao-Ying Lin, Member, IEEE, and Wen-Guey Tzeng, Member, IEEE, “A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding”.