BIOINFO Security Informatics
ISSN: 2249-9423 & E-ISSN: 2249-9431, Volume 2, Issue 2, 2012, pp.-53-57.
Available online at http://www.bioinfo.in/contents.php?id=344

SECURITY IN CLOUD COMPUTING

GAWANDE Y.V.[1], AGRAWAL L.S.[2], BHARTIA A.S.[3] AND RAPARTIWAR S.S.[4]

Department of Computer Science and Engineering, J.D.I.E.T, Yavatmal, India
*Corresponding Author: Email- yashashree.gawande@gmail.com

Received: March 15, 2012; Accepted: April 12, 2012

Abstract- In cloud computing the security facts including storage security, data security, and network security and how to handle encrypted data. Then we select some topics and describe them in more detail. In particular, we discuss a scheme for secure third party publications of documents in a cloud. Next we discuss secure federated query processing with map Reduce and Hadoop. Next we discuss the use of secure co-processors for cloud computing. Third we discuss XACML implementation for Hadoop. We believe that building trusted applications from untrusted components will be a major aspect of secure cloud computing.
Keywords- Cloud Computing, Security, Hadoop, Untrusted components, Federated Query.

Citation: Gawande Y.V., et al. (2012) Security in Cloud Computing. BIOINFO Security Informatics, ISSN: 2249-9423 & E-ISSN: 2249-9431, Volume 2, Issue 2, pp.-53-57.

Introduction

There is a critical need to securely store, manage, share and analyze massive amounts of complex data to determine patterns and trends in order to improve the quality of healthcare better safeguard the nation and explore alternative energy. Because of the critical nature of the applications, it is important that clouds be secure. The major security challenge with clouds is that the owner of the data may not have control of where the data is placed. This is because if one wants to exploit the benefits of using cloud computing, one must also utilize the resource allocation and scheduling provided by clouds. Therefore there is need to safeguard the data in the midst of untrusted processes.

The emerging cloud computing model attempts to address the explosive growth of web-connected devices, and handle massive amounts of data. Google has now introduced the MapReduce framework for processing large amounts of data on commodity hardware. Apache's Hadoop distributed file system (HDFS) is emerging as a superior software component for cloud computing combined with integrated parts such as MapReduce. The need to augment human reasoning, interpreting, and decision making abilities have resulted in the emergence of the Web, which is an initiative that attempts to transform the web from its current, merely human-readable form, to a machine processable form. This in turn has resulted in numerous social networking sites with massive amounts of data to be shared and managed. Therefore there is urgently need a system that can scale to handle a large number of sites and process massive amounts of data. However, it provided state of the art systems utilizing HDFS and MapReduce are provide security mechanisms to protect sensitive data.

Due to the extensive complexity of the cloud, it will be difficult to provide a holistic solution to securing the cloud at present. Therefore there is needed to make increment enhancements to securing the cloud that ultimately results in a secure cloud. In particular, a secure cloud consisting of hardware (includes 800TB of data storage on a mechanical disk drive, 2400 GB of memory and several commodity computers), software (includes Hadoop) and data. So cloud system:

1) support efficient storage of encrypted sensitive data,
2) store, manages and query massive amounts of data,
3) support fine grained access control and
4) support strong authentication.

It describes, approach to securing the cloud. In session 2, We will give an overview of security issues for Cloud. In session 3, We will discuss secure third party publication in clouds. In session 4,

We will discuss how secure data storage for cloud. Session 5, We will discuss Hadoop for cloud computing. And in session 6, there is SPRQL query optimization. There is conclusion in session 7.

Overview of Security

There are numerous security issues for cloud computing as it encompasses many technologies including networks, databases, operating systems, virtualization, resource scheduling, transaction management, load balancing and memory management. Therefore, security issues for many of these systems and technologies are applicable to cloud computing. For example, the network that interconnects the systems in a cloud has to be secure. For example, mapping the virtual machines to the physician machines has to be carried out securely. Data security involves encrypting the data as well as ensuring that appropriate policies are enforced for data sharing. In addition, resource allocation and memory management algorithms have to be secure.

We will focus only on some aspects of secure cloud computing. One is to efficiently store the data in foreign machines. Another is to query encrypted data as much of the data on the cloud may be encrypted.

By using Hadoop distributed file system for virtualization and applying security for Hadoop which includes a query optimization. In addition there is investigation for secure query processing on clouds over Hadoop.

Third Party Publication Applied to Cloud

Cloud computing facilitates storage of data at a remote site to maximize resource utilization. As a result, it is critical that this data be protected and only given to authorized individuals. This essentially amounts to secure third party publication of data that is necessary for data outsourcing as well as external publications. Here the data is represented as an XML document.

Many of the documents on the web are now represented as XML documents. First there is access control framework proposed in [1] and then secure third party publication. In the access control framework proposed in [1], security policy is specified depending on user roles and credentials. Users must possess the credentials to access XML documents. The credentials depend on their roles. For example, a professor has access to all of the details of students while a secretary only has access to administrative information. XML specifications are used to specify the security policies. Access is granted for an entire XML document or portions of the document. Under certain conditions, access control may be propagated down the XML tree.

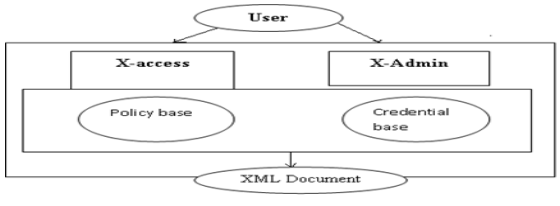

Fig. 1- Access Control Framework

For example, if access is granted to the root, it does not necessarily mean access is granted to all the children. One may grant access to the DTDs and not to the document instances. One may grant access to certain portions of the document.

For example, a professor does not have access to the medical information of students while he has access to student grade and academic information. Design of a system for enforcing access control policies are also described in [1]. Essentially the goal is that the user is authorized to see the XML views as specified by the policies.

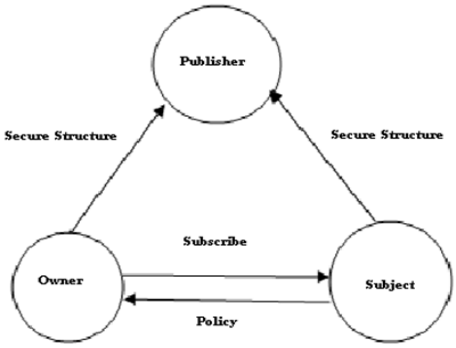

Fig. 2- Secure Third Party Publication

In Figure2. The idea is to have untrusted third party publishers. The owner of a document specifies access control polices for the subjects. Subjects get the policies from the owner when they subscribe to a document. The owner sends the documents to the Publisher. When the subject requests a document, the publisher will apply the policies relevant to the subject and give portions of the documents to the subject. Now, since the publisher is untrusted, it may give false information to the subject. Therefore, the owner will encrypt various combinations of documents and policies with his/her private key. Using Merkle signature and the encryption techniques, the subject can verify the authenticity and completeness of the document for secure publishing of XML documents. In the Cloud environment, the third party publisher is the machine that stored the sensitive data in the cloud. This data has to be protected and the techniques above have to be applied to that authenticity and completeness can be maintained.

Secure Data Storage For Cloud

Since data in the cloud will be placed anywhere, it is important that the data is encrypted. By using secure co-processor parts cloud to enable efficient encrypted storage of sensitive data. By embedding a secure co-processor (SCP) into the cloud infrastructure, the system can handle encrypted data efficiently in Figure 3.

Basically, SCP is a tamper-resistant hardware capable of limited general-purpose computation. For example, IBM 4758 Cryptographic Coprocessor [IBM04] is a single-board computer consisting of a CPU, memory and special-purpose cryptographic hardware contained in a tamper-resistant shell; certified to level 4 under FIPS PUB 140-1. When installed on the server, it is capable of performing local computations that are completely hidden from the server. If the tampering is detected then the secure co-processor clears the internal memory. Since the secure coprocessor is tamper-resistant, one could be tempted to run the entire sensitive data storage server on the secure co-processor. Pushing the entire data storage functionality into a secure co processor is not feasible due to many reasons.

First of all, due to the tamper-resistant shell, secure co- processors have usually limited memory (only a few megabytes of RAM

and a few kilobytes of non volatile memory) and computational power. Performance will improve over time, but problems such as heat dissipation/power use (which must be controlled to avoid disclosing processing) will force a gap between general purposes and secure computing. Another issue is that the software running on the SCP must be totally trusted and verified. This security requirement implies that the software running on the SCP should be kept as simple as possible [4].

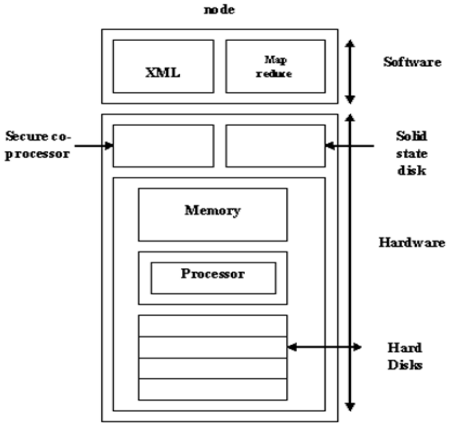

Fig. 3- Parts of propose instrument

So this hardware help in storing large sensitive data sets. By encrypting the sensitive data sets using random private keys and to alleviate the risk of key disclosure, using tamper-resistant hardware to store some of the encryption/decryption keys. (i.e., a master key that encrypts all other keys). Since the keys will not reside in memory unencrypted at any time, an attacker cannot learn the keys by taking the snapshot of the system. Also, any attempt by the attacker to take control of (or tamper with) the co-processor, either through software or physically, will clear the co-processor, thus eliminating a way to decrypt any sensitive information. This framework will facilitate (a) secure data storage and (b) assured information sharing. For example, SCPs can be used for privacy preserving information integration which is important for assured information sharing.

We have conducted on querying encrypted data. With Secure Multipart Computation SMC protocols, one knows about his own data but not his partner's data since the data is encrypted. However, operations can be performed on the encrypted data and the results of the operations are available for everyone. One drawback of SMC is the high computation costs.

Query Processing With Hadoop

Overview of HADOOP

A major part of system is HDFS which is a distributed Java-based file system with the capacity to handle a large number of nodes storing petabytes of data. Ideally a file size is a multiple of 64 MB. Reliability is achieved by replicating the data across several hosts. The default replication value is 3 (i.e., data is stored on three nodes). Two of these nodes reside on the same rack while the other is on a different rack. A cluster of data nodes constructs the file system. The nodes transmit data over HTTP and client's access data using a web browser. Data nodes communicate with each other to regulate, transfer and replicate data.

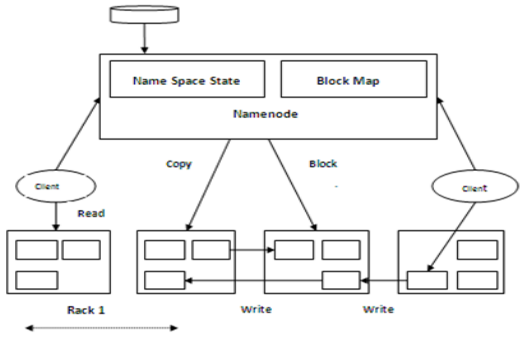

Fig. 4- HADOOP distributed file structure. (HDFS architecture)

HDFS architecture is based on the Master-Slave approach in Figure 4. The master is called a Namenode and contains metadata. It keeps the directory tree of all files and tracks which data is available on which node across the cluster. This information is stored as an image in memory. Data blocks are stored in Datanodes. The namenode is the single point of failure as it contains the metadata. So, there is optional secondary Namenode that can be setup on any machine. The client accesses the Namenode to get the metadata of the required file. After getting the metadata, the client directly talks to the respective Datanodes in order to get data or to perform IO actions. On top of the file systems there exists the map/reduce engine. This engine consists of a Job Tracker. The clients applications submit map/reduce jobs to this engine. The Job Tracker attempts to place the work near the data by pushing the work out to the available task taker nodes in the cluster.

Inadequacies of Hadoop:- Current systems utilizing Hadoop have the following limitations.

a) No facility to handle encrypted sensitive data: Sensitive data ranging from medical records to credit card transactions need to be stored using encryption techniques for additional protection. Currently HDFS does not perform secure and efficient query processing over encrypted data.
b) Web Data Management: There is a need for viable solutions to improve the performance and scalability of queries against web data such as RDF (Resource Description Framework). The number of RDF datasets is increasing. The problem of storing billions of RDF triples and the ability to efficiently query them is yet to be solved. At present, there is no support to store and retrieve RDF data in HDFS.
c) No fine grained access control: HDFS does not provide fine grained access control. There is some work to provide access control lists for HDFS [5]. For many applications such as assured information sharing, access control lists are not sufficient and there is a need to support more complex policies.
d) No strong authentication: A user who can connect to the Job Tracker can submit any job with the privileges of the account used to set up the HDFS. Future versions of HDFS will support network authentication protocols like Kerberos for user authentication and encryption of data transfers [5].

Sparql Query Optimization

While the secure co-processors can provide the hardware support

to query and store the data, there is need to develop a software system to store, query, and mine the data. More and more applications are now using semantic web data such as XML and RDF due to their representation power especially for web data management. Therefore, by exploring ways to securely query semantic web data such as RDF data on the cloud. By using several software tools that are available to help in the process including the following:

Jena: Jena is framework which is widely used for solving SPARQL queries over RDF data. But the main problem with Jena is scalability. It scales in proportion to the size of main-memory. It does not have distributed processing. However, by using Jena in the initial stages of our preprocessing steps.[6]

Pellet: Pellet to reason at various stages. So by doing real time query reasoning using pellet libraries [6] coupled with Hadoop's map-reduce functionalities.

Pig Latin: Pig Latin is a scripting language which runs on top of Hadoop [7]. Pig is a platform for analyzing large data sets; Pig's language, Pig Latin, facilitates sequence of data transformations such as merging data sets, filtering them, and applying functions to records or groups of records. It comes with many built-in functions but any can also create our own user-defined functions to do special-purpose processing. Using this scripting language, by avoid writing our own map-reduce code; rely on Pig Latin's scripting power that will automatically generate script code to map-reduce code.

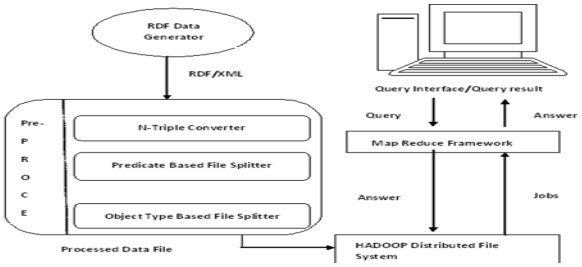

Fig. 5- Architecture of SPARQL query optimization

There are a number of steps to preprocess and query RDF data in Fig. 5. With this proposed part, researchers can obtain results to optimize query processing of massive amounts of data [9]. Below there is discussion on the steps involved in this part.

Pre-processing: Generally, RDF data is in XML format. In order to execute a SPARQL query, by proposing some data pre-processing steps and store the pre-processed data into HDFS. There is an N-triple Convertor module which converts RDF/XML format of data into N-triple format as this format is more understandable. There is use of Jena framework as stated earlier, for this conversion purpose.

In Predicate Based File Splitter module, by split all N-triple format files based on the predicates. Therefore, the total number of files for a dataset is equal to the number of predicates. In the last module of the pre-processing step, by further dividing predicate files on the basis of the type of object it contains. So, now each predicate file has specific types of objects in it. This is done with the help of the Pellet library. This pre-processed data is stored into Hadoop.

Strong Authentication

Hadoop does not authenticate users. This makes it hard to enforce access control for security sensitive applications and makes it easier for users to circumvent file permission checking done by HDFS. To address these issues, the open source community is actively using to Kerberos protocols with Hadoop [10]. On top of the proposed Kerberos protocol, for some assured information applications, there may be a need for adding simple authentication protocols to authenticate with secure coprocessors. For this reason, one can add a simple public key to system so that users can independently authenticate with secure co-processors to retrieve secret keys used for encrypting sensitive data. There is use open source public key infrastructure such as the Open PKI implementation for system.

Conclusion

There are several security challenges including security aspects. There believes that due to the complexity of the cloud, it will be difficult to achieve end-to-end security. So security issues for cloud are important. These issues include storage security, data security, network security and application security. The main goal is to securely store and manage data that is not controlled by the owner of the data. Then there is focused on specific aspects of cloud computing. In particular, taking a bottom up approach to security where working on small problems in the cloud, that there is one hope to solve the larger problem of cloud security. Firstly, how to secure documents that may be published in a third party environment. Next how to secure co-processors may be used to enhance security. Finally, Hadoop environment as well as in secure federated query processing with SPARQL using MapReduce and Hadoop.

Acknowledgment

We are grateful to Prof. M.V. Sarode, Associate Professor and Head, Computer Science & Engineering Department and Dr. A. W. Kolhatkar, Principal, Jawaharlal Darda Institute of Engineering & Technology for their excellent support during my work. Last, but not least We would like to thank all professors and lecturers, and members of the department of Computer Science and Engineering, Jawaharlal Darda Institute of Engineering & Technology, Yavatmal for their generous help in various ways for the completion.

References

[1] Bertino E., et al. (2002) *Access Control for XML Documents Data and Knowledge Engineering*, 43, 03.
[2] Bertino E. et al. (2004) *IEEE Transactions on Knowledge and Data Engineering*.
[3] Smith S.W. and Weingart S.H., *Building a high-performance, programmable secure coprocessor*.
[4] Hamlen K.W., Momsett G. and Schneider F.B. (2006) *The ACM SIGPL4V Workshop on Programming Languages and Analysis for Security (PLAS).*
[5] Zhang K., *Adding user and service-to-service authentication to Hadoop*.
[6] Hamlen (2009) *IEEE Intelligence and Security Informatics Conference pSI).*

[7] Gates F., Natkovich O., Chopra S., Kamath P., Narayanamurthy S.M., Olston C., Reed B., Srinivasan S. and Srivastava U., *Pig Experience.*
[8] Newman A., Hunter J., Li Y.F., Bouton C. and Davis M., *A Scale-Out RDF*.
[9] Hurtado C.A., Poulovassilis A. and Wood P.T. (2006) *International Semantic Web Conference*.
[10] Zhang K., *Adding user and service-to-service authentication to Hadoop*.
[11] Kelvin Hamlen. Murat Kantarcioglu, Latifur Khan, and Bhavani Thuraisingham *Seurity.*