A Client Centric Hadoop Ecosystem

Raj Kushal Ananth Kumar, Pooja Singh, Naman Pradhan, Afroz Pasha

***Abstract* - Various public cloud service offerings such as Google App Eng, Amazon Web Services, Microsoft Azure, etc. are increasingly gaining popularity as a means to perform network, storage and compute operations. Also the Hadoop framework is rapidly becoming the industry standard for most organizations due to its scalability, fault tolerant and cost effective design. Hence deploying Hadoop on the cloud will allow leveraging extremely elastic and flexible operations all while being cost and time effective. However various cloud attacks can bypass the current Hadoop security mechanisms, and threaten the confidentiality of the client's data and the overall system. In this paper we propose a Hadoop System that maintains the privacy and security of the information stored on the cloud through Client side validation and encryption along with a more resilient public cloud-based Hadoop model.**

Keywords* - client encryption; *Hadoop; Public cloud; Resiliency; Security and privacy;

I. INTRODUCTION

Cloud computing[1] is a model for enabling ubiquitous, convenient, on-demand access to a shared pool of configurable computing resources. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers.[2] It relies on sharing of resources to achieve coherence and economies of scale. It is accepted as fact that there are a multitude of technical reasons why a company could move to the cloud, these reasons include [3] reduced IT costs, Scalability, Business continuity, Location independence etc. Hadoop [4] is an open source, java-based framework that supports the processing of large data sets in a distributed computing environment. It was originally conceived on the basis of Google's Map Reduce. The main components of the Hadoop Eco System are Hadoop Distributed File System and Map Reduce. HDFS is the cloud storage system for the data and Map Reduce is used for processing and analyzing the data.

In recent times implementing a Hadoop eco-system on the cloud has become a trending topic. Companies such as Amazon, VMWare and Rackspace all announced products or services marrying Hadoop with the cloud [5]. The benefits include lowering the cost of innovation, procuring large scale resources quickly, Running services and operations closer to the data and simplifying Hadoop Operations. However Hadoop was never originally designed to run on a public cloud as the traditional Hadoop security mechanisms cannot take into measure the added dangers of resource sharing at the hardware as well as the software level. Most cloud attacks involving a Hadoop framework implementation aims at attacking the infrastructure as a service (IaaS) cloud service. In this paper we propose to make the current system more resilient by:

a) Implementing a Robust architectural model called the CC Hadoop model

b) Creating a client centric block token validation mechanism called CCHadoop block token

The client in many cases would not like to leave his private data on the cloud without some form of encryption; hence there is a need for establishing trust between the client and the NameNode of the Hadoop System. To achieve this we propose a system by which we first authenticate the user, so only a valid user can send requests to the NameNode and implement an encryption technique on the data. We propose to use Map Reduce for the encryption/decryption process so as to ensure that the performance and scalability of the system is not affected.

II. IAAS THREAT ANALYSIS

In a public cloud we have multiple clients that would be sharing resources such as hardware resources as well as software resources. IaaS also by definition includes threats related to PaaS and SaaS. It also includes many other threats because of the nature of IaaS offering. The number one threat for the consumer of an IaaS offering are vulnerabilities in the underlying operating system or services that are running on it. Linux (and variants) and Windows based OSs are the main options you have in Public IaaS offerings. While other OSs may be available, Linux and Windows are +90% of the market. Both of these OSs and services that run on them have (and will continue to have) vulnerabilities. OS and service vulnerabilities are publicized through many outlets, and in many instances exploits are publicly available.

By definition IaaS resources are remote, and thus we need some type of remote management mechanism. The most popular mechanisms to accomplish the remote functions are:

a) Virtual Private Networks

b) Remote Desktops

c) Remote Shell

d) Web Console User Interface

Revised Version Manuscript Received on October 29, 2015.

RajKushal Ananth Kumar, Manhattan Associates and aims to pursue his masters in computer science. Nitte Meenakshi Institute of Technology, Govindpura, Gollahalli, Yelahanaka, Bengaluru, (Karnataka). India.

Pooja Singh, Capgemini and aims to pursue her Masters in Computer Science. Nitte Meenakshi Institute of Technology, Govindpura, Gollahalli, Yelahanaka, Bengaluru, (Karnataka). India.

Naman Pradhan, Larsen and Tubro. Cloud Computing and Databases, Nitte Meenakshi Institute of Technology, Govindpura, Gollahalli, Yelahanaka, Bengaluru, (Karnataka). India.

Afroz Pasha, Assistant Professor, Nitte Meenakshi Institute of Technology, Govindpura, Gollahalli, Yelahanaka, Bengaluru, (Karnataka). India.

Poor credentials are a threat affecting all the remote management solutions mentioned above. Weak authentication, due to poor credentials,

is one of the main reasons IaaS can get compromised [6] when connected to the Internet through public clouds. Also Hadoop administrators may forget to delete the secret keys which are used by Node Managers and DataNodes to authenticate themselves to each other. This can be through compromising the block token that is used between the Node Manager and the DataNodes. Also an attacker can compromise the inherent weaknesses of the architecture of the public cloud through vulnerabilities in hypervisors [7] [8], malicious VMs [9] or through side-channel attacks [10] when instances share the same physical server.

In short, when we implement Hadoop on a public cloud using IaaS, attackers can launch internal cloud attacks to compromise the Hadoop security systems, and steal sensitive information from the client.

III. CCHADOOP MODEL

A. Overview of CCHADOOP

To improve the current Hadoop framework, when deployed on a public facing cloud, we design a more resilient CCHadoop model that focuses on two main principles: enhancing the isolation of high risk and priority Hadoop components through a CCHadoop Infrastructure Model and overcoming the implicit vulnerability in the traditional Hadoop Block token system by introducing a novel CCHadoop Bock token mechanism. When a Hadoop component such as a DataNode is compromised, strong isolation levels can prevent any damage to the core Hadoop components such as the NameNode, Application Masters, Job Clients, and Kerberos which run in a secure zone. Since some Hadoop components are deployed on a large number of VM's, such as DataNode, Node Manager, Resource Manager and Container, there is a large possibility being attacked than other Hadoop components since they are in the public cloud where they are under threat due to resource sharing from other cloud tenants.

Also when we do not have our confidential data hosted on a remote server as in the case of a public cloud, the NameNode and the Data Node have the control over the data and not the client, and so there is need to establish some level of trust between the client and the Hadoop system. To achieve this, we have used client side encryption to protect the clients' sensitive data through AES-256 bit encryption as well as authentication through public-private key pairs.

B. CCHadoop Infrastructure Model

The infrastructure of CC Hadoop Model is divided into three zones. The first zone is the public cloud for components that need low security, the other zone being the private cloud for systems that require high security and a third zone called the maintenance zone.

1) Public cloud - This zone is on the public cloud and is virtually divided into two subnets which would be the public facing subnet and the private subnet. The reason this zone is on the public cloud is to reduce costs. The Hadoop components running in this zone are the DataNode, Node Manager, Resource Manager and the Container.

- The public facing subnet - This subnet is the subnet that would have services like the Web Servers, Load Balancers etc. These services are configured to filter inbound traffic. For example if the public subnet contains the web servers, the inbound traffic would be filtered to port 80 and 443 for HTTP and HTTPS. This subnet does not have any filtering on the outbound traffic and can directly access the internet gateway. The main reason to isolate this region is to provide an extra layer of security to the servers running Hadoop.
- The private subnet - Private subnet in the public cloud contains the DataNodes and the resource manager. This zone is isolated from the internet in the sense that any inbound traffic can access this subnet only if it is allowed to pass through the PAT server that connects the two subnets in the public cloud. This subnet can access the other zone of the CC Hadoop model through hardware based tunneling.
- PAT (Port Address Translation) Server - A PAT server is a transit device used in this model to maintain a strict control over the inbound and outbound traffic meant for the private subnet. A PAT instance does a port address translation and gives access to the servers in the public subnet to access specified ports. This server makes sure that any traffic that goes into the private subnet is from the public subnet only and rejects traffic from IP's outside the range of the public subnet meaning that the internet cannot access the private subnet. The source destination check is disabled for this server since it is not a destination for any of the subnets but a passage for the traffic.

2) Private Cloud - Private cloud is a high security region physically separate from all the other subnets. This contains the core of the CC Hadoop Model, the NameNode. This zone can access only the private subnet in the public cloud by means of tunneling using a CISCO 2611 XM router. Apart from the NameNode this zone also includes the Hadoop Application Masters, Job Clients and the Kerberos authentication server.

3) Maintenance Zone - Since the private subnet cannot be accessed by the internet, in case of maintenence issues and versioning it would be a problem to access the servers inside the private subnet. Using this maintenence zone, private subnet can accept maintenance requests from the maintenance server and not directly from the internet. *The diagrammatic representation of the CCHadoop Architecture is as shown in Fig 1.*

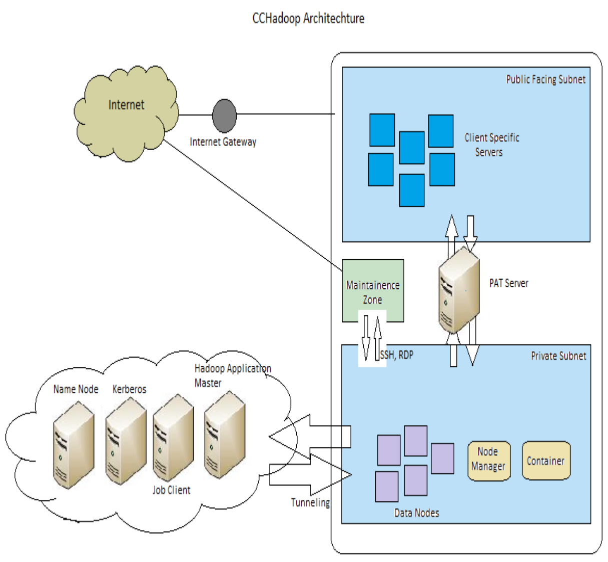

Figure 1: CCHadoop Architecture

C. Establishing Trust

In Hadoop system, the NameNode and the DataNodes handle the data for its storage and access when requested by the user. The user has no control over handling the data and so, there is a need for the establishment of trust between the NameNode and the User. Not every User should be given the privilege of accessing and storing the data. The User should authenticate himself to the NameNode before he is granted access. Hashing techniques are implemented to achieve the authentication. The hashing technique used in this system is SHA-256. The user authenticates himself to NameNode by sending a hash function. NameNode then generates a hash function and compares it with the hash function sent by the User In Hadoop, the NameNode and the Data Node handle all the storage and access of the client data whenever required. Thus the client has no control over the handling of his confidential data and so, there is a need to establish some mechanism that builds trust between Hadoop and the Client. Thus we implement a system by which the user should first authenticate himself to the NameNode before he is granted access to the Data Node. An RSA algorithm is used to generate a public - private key pair. The client has access to his private key while the Name Node stores the public keys of all users in a hash map. Also the client stores his data on the Data Node by first encrypting the data with his private key using an AES-256 bit encryption scheme. The validation and encryption procedure is carried out in the below two steps:

a) The Client first has to validate himself to the Name Node by providing an encrypted message consisting of his private key and his credentials. This message is decrypted by the Name Node using the public key stored and the client credentials are extracted. If the validation is successful then the NameNode generates the CCBlock token.

b) After the CCBlock tokens are accepted by the corresponding DataNodes, the user runs a map reduce job on the data stored on the DataNodes. This map reduce job decrypts the data that was encrypted previously by the Client using an AES-256 bit encrytion technique.

D. CCHadoop Runtime Model

The NameNode manages the Hadoop file system namespace and has a secret key to generate Block Tokens to be used between the NameNode and the DataNodes to authenticate themselves. If a malicious user or an attacker can gain access to this key, he will be able to create his own block tokens to access data from any DataNode in the Hadoop System. Hence we design a CCHadoop Block Token to overcome this implicit vulnerability. CCHadoop uses this secret key to perform authentication, but the NameNode here generates different secret keys for every DataNode. The CCHadoop Block Token then uses AES-256 bit encryption technique to encrypt the CCHadoop block tokens content. Hence the CCHadoop block tokens content remains protected during transfer from a NameNode to DataNode. The CCHadoop Block token contains, along with the standard Hadoop Block token, the private key of the client. This private key is then used by the client to decrypt his confidential data stored on the DataNode using an AES-256 bit decryption technique.

The main advantage of such a model is that even if the CCHadoop Block token is intercepted by a hacker or a malicious user, then he would not be able to deduce the network topology of the data in the HDFS. Also since we perform the decryption by leveraging the existing map-reduce feature of Hadoop, we do not subject the system to any additional overhead.

E. CCHadoop Block Token

In CCHadoop model, a Name Node uses a unique Block token to authenticate with the DataNodes, we call this Block Token as the CCHadoop Block Token. This CC Hadoop block token is generated by a secret key given by the NameNode. Each DataNode is generated a unique secret key by the NameNode. After the client is authenticated with his private key, for all the Data Nodes that he wishes to access a CCHadoop Block token is generated. The CCHadoop Block Tokens generation format is as shown below:

CCBTID = {ED, UID, AM, BID, BPID, CIP, KeyID, PK} (1)
CCBT = AES {CCBTID} (2)

Where ED stands for expiration date of Block Token, UID stands for User ID, AM means Access Mode (e.g. write, copy, replace, read), BID defines which block needs to be operated on, BPID is used to represent block pool ID, CIP is the HDFS Client’s IP address, KeyID is used to identify which key is used, PK represents the clients Private Key, CCBTID stands for CCHadoop Block Token ID which defines all necessary information for an HDFS access request, AES represents the Advanced Encryption Standard (AES) encryption function we used to encrypt the plaintext of the Block Token, and CCBT represents CCHadoop Block Token. The symmetric unique secret key is used in AES function in format (2) to generate the required CCHadoop Block Tokens.

This CCHadoop Block Token is sent to the respective DataNode it needs to access. The DataNode then uses its symmetric secret key to decrypt the CCBT and obtain the CCBTID. From this CCHadoop Block Token, the KeyID is compared with the KeyID stored on the DataNode and if the validation is successful the DataNode allows the Client request to access it.

Next an AES-256 decryption map-reduce job is run on the data stored on the DataNode using the private key. This gives us the original contents of the data.

IV. CCHADOOP IMPLEMENTATION & EXPERIMENTS

To verify the CCHadoop system effectiveness and efficiency in a real time environment, we implemented the proposed architecture to host a web application with static and dynamic content accessed by multiple clients deployed on a commercial public cloud. The CCHadoop model was implemented on Amazon EC2 server for this experiment. The OS was an m4.xlarge configured with the predefined Amazon ami "Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-d05e75b8" on which VMware was installed. The CPU had 4 cores and had a 2.4 GHz Intel Xeon® E5-2676 v3 (Haswell) processor supported for enhanced networking. CCHadoop was set up on 8 VM's with each having 8 GB disk space. One VM was the master node with 6 GB memory. 6 VM's with 8 GB memory each. A Kerberos server with a memory of 2 GB. The NameNode, the Resource Manager, and the Job Client ran on master VM and a DataNode and Node Manager ran on the each slave nodes. Using our model the web application could be accessed successfully within 1.8 seconds in average access identification to exactly cross-examine the subject identity. As a result the proposed Hadoop cloud computing model performed very well when it was deployed in this local area.

We developed two attack scenarios: The HDFC attack and the block token attack. Each attack was simulated on both Hadoop and CCHadoop separately. The HDFC attack tried to take advantage of a compromised NameNode in the public cloud and the Block Token attack we tried to read a Block Token of a DataNode by a malicious user from another DataNode.

When we tested the resiliency of the architecture when a DataNode was compromised. In this scenario, one DataNode in the private part of the public cloud was accessible to a malicious user, direct access was given to a user through the PAT server to check if this user can in any way access the NameNode which was in the private cloud. This user tried to exploit the vulnerabilities of the architecture but the NameNode remained safe from that user due to the tunneling. The ports in this public network remained unaware of the private network and could not access the NameNode what so ever. In case of the regular Hadoop model where the NameNode and the DataNodes are in the same network, when this experiment was done, this user was successful in fetching data which it could not in case of the fault-tolerant architecture of CCHadoop.

Also, we evaluated CCHadoop Block Token performance by creating a malicious DataNode to read an HDFS block of another DataNode. The attack code constructed a desired HDFS block and then used the keys shared between the NameNode and the DataNode to generate a Block Token. Using the traditional Hadoop model, our experiments show that an attacker can easily gain access to the data stored on ant DataNode of the Hadoop system as the Block Token for all the DataNodes are the same, whereas when the same scenario was simulated in the CCHadoop model, on account of the encrypted CCHadoop Block Token, and the unique symmetric key Block Token generation mechanism, the target DataNode rejected the token request and the attack failed.

Finally, we did not find any significant difference between the performance of the Hadoop model and the CCHadoop model. The CCHadoop made use of low overhead algorithms to generate the CCHadoop Block Tokens. Also since we performed the encryption/decryption of the client data by leveraging the existing map-reduce feature of Hadoop, we do not subject the system to any additional overhead.

V. CONCLUSIONS

After performing a thorough analysis on the various IaaS cloud attacks a Hadoop implementation on the cloud is exposed to, we have designed and implemented the CCHadoop model. We have shown how we can enhance the isolation levels of core Hadoop processes in the CCHadoop Infrastructure model. Also we have shown how we can make the overall model more resilient and fault tolerant with the CCHadoop Block Token mechanism. Finally to protect the confidentiality of the clients data stored on the Data Nodes, we have employed a client side Encryption and validation scheme.

REFERENCES

1. HongBo Zhou. Cloud computing: technology, application,standar, Electronic Industry Press.2011
2. Al-Fares M et al (2008) A scalable, commodity data center network architecture. In: Proc SIGCOMM
3. Cloud Computing: benefits, risks and recommendations for information security. D Catteddu - Web Application Security, 2010 - Springer
4. Apache Hadoop. http://hadoop.apache.org/, 2012
5. Inforchimps,"Cloud::hadoop,"http://www.infochimps.com/infochimpscloud /cloud- services/cloud-hadoop/ Accessed in Oct. 2015.
6. D. Zissis and D. Lekkas, Addressing cloud computing security issues, Future Generation Computer Systems, vol. 28, no. 3, pp. 583 – 592, 2012.
7. S. Advisory, Xen pv kernel decompression multiple vulnerabilities, http://secunia.com/advisories/44502/. Accessed in November 2014.
8. F. Rocha and M. Correia, Lucy in the sky without diamonds: Stealing confidential data in the cloud, in Dependable Systems and Networks Workshops (DSN-W), 2011 IEEE/IFIP 41st International Conference on. IEEE, 2011, pp. 129–134.
9. S. Bugiel, S. Nurnberger, T. P¨ oppelmann, A.-R. Sadeghi, and T. Schnei-¨ der, Amazonia: when elasticity snaps back, in Proceedings of the 18th ACM conference on Computer and communications security. ACM, 2011, pp. 389–400.
10. T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds, in Proceedings of the 16th ACM conference on Computer and communications security. ACM, 2009, pp. 199212.

AUTHORS PROFILE

RajKushal Ananth Kumar, completed his BE in 2015, he is presently working in Manhattan Associates and aims to pursue his masters in computer science. His areas of interest include BigData, Hadoop, Cloud Computing and Machine learning.

Pooja Singh, completed her BE in 2015, she is presently working in Capgemini and aims to pursue her Masters in computer Science. Her areas of interest include BigData, Hadoop, Cloud Computing and Artificial Intelligence.

Naman Pradhan, completed his BE in 2015, He is presently working in Larsen and Tubro. His areas of interest include BigData, Hadoop, Cloud Computing and Databases.

Afroz Pasha, completed his BE in 2006 and MTech in 2012. He is presently working as Assistant Professor in Nitte Meenakshi Institute of Technology Bengaluru. His areas of interest include Cloud computing security and databases.