Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 9, September 2013, pg.1 – 7

RESEARCH ARTICLE

INTRUSION DETECTION ON CLOUD APPLICATIONS

A. Venkat Reddy[1], K. Sharath Kumar[2], V. Hari Prasad[3]

[1]M.Tech (CSE), Sphoorthy Engineering College, JNTU, Hyderabad, India

[2]Associate Professor, Sphoorthy Engineering College, JNTU, Hyderabad, India

[3]Head of the Department (CSE & IT), Sphoorthy Engineering College, JNTU, Hyderabad, India

[1] *v.anugureddy@gmail.com*

Abstract— Cloud computing has grown from being a promising business concept to one of the fastest growing segments of the IT industry. So, cloud environment always remains vulnerable to attacks. The framework serves as an excellent platform for making cloud services intrusion tolerant. The feasibility of the framework has been tested by making cloud's Infrastructure as a Service (IaaS) and Data Storage Service intrusion tolerant. The proposed framework has been validated by integrating

Intrusion Tolerance via Threshold Cryptography (ITTC) mechanism in the simulated cloud's IaaS. For this, the data centre authentication key is distributed among the hosts using Shamir Secret Sharing algorithm. Performance of the new simulated service model is measured using various performance metrics such as total execution time, intrusion detection time, recovery time, number of cloudlets etc. It involves, using proposed Cloud Intrusion Tolerance framework for securing cloud Data Storage. The correctness of user's data is ensured by using erasure-correcting code in the file distribution preparation to provide redundancy parity vectors. Performance analysis using erasure-correcting code for securing data storage is also done. We are also implementing the different networks such as SHIM (System Health and Intrusion Monitoring) is used as an exemplary host-based IDS to validate our approach, DRCP (Dynamic Registration and Configuration Protocol) is an auto configuration protocol in mobile AdHoc networks, OLSR (Optimized Link State Routing) protocol is a proactive, table-driven routing protocol in MANETs.

Keywords: - Cloud Computing; Data Storage; DRCP; Framework; Intrusion Tolerance; OLSR; Reed-Solomon Encoding; Shamir Secret Sharing; SHIM; Security; Threshold Cryptography

I. INTRODUCTION

In the last few years, cloud computing has grown from being a promising business concept to one of the fastest growing segments of the IT industry. Moving critical applications and sensitive data to a public and shared cloud environment is a major concern for corporations that are moving beyond their data centre's network perimeter defense due to various security issues in cloud environment. Intrusion Tolerance in Cloud Computing is a fault tolerant design approach to defend cloud infrastructure against malicious attacks. Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. Cloud enables the consumers of the technology to think of computing as effectively limitless, of minimal cost, and reliable, as well as not be concerned about how it is constructed, how it works, who operates it, or where it is located. Some examples of emerging cloud computing infrastructure are Microsoft Azure, Amazon EC2, Google App Engine, and Aneka.

© 2013, IJCSMC All Rights Reserved

For secure data storage, we have used erasure-correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee data dependability. Erasure-correcting code i.e. Reed-Solomon codes are used to provide intrusion tolerance for data servers in the cloud. Performance of the proposed intrusion tolerant data storage is measured in terms of Storage time, Intrusion detection/localization time, recovery time etc. Performance analysis shows that the proposed scheme is highly efficient and resilient against malicious data modification attack. The rest of the paper includes following structure, Related Work provides a brief summary of the related work in this area. In Frame Work, we propose our framework. Frame Work Validation gives the validation of our proposed framework.

II. RELATED WORK

An intrusion detection and tolerance is a system that is capable of self-diagnosis, repair, and reconfiguration while continuing to provide a correct service to legitimate users in the presence of intrusions.

A. Security Framework for Cloud Environment:

Security framework for cloud environment consists of,

a. Secure Channels and Envelopes includes Communicating in a secure way and dispatching information in a secure way

b. Authentication includes ensuring what we deal with is genuine: end-users, data, servers, etc.

c. Protection and Authorization

i. Protecting resources from unauthorized access

ii. Ensuring the users are authorized to do just what they should

d. Auditing and Intrusion Detection

i. Following the system execution for a posterior analysis

ii. Detecting anomalous usage in runtime

B. Cloud Simulation Toolkit:

Evaluating the performance of Cloud provisioning policies, application workload models, and resources performance models in a repeatable manner under varying system and user configurations and requirements is difficult to achieve. To overcome this challenge, CloudSim is proposed: an extensible simulation toolkit that enables modeling and simulation of Cloud computing systems and application provisioning environments. The CloudSim toolkit supports both system and behavior modeling of Cloud system components such as datacenters, virtual machines (VMs) and resource provisioning policies. It implements generic application provisioning techniques that can be extended with ease and limited efforts. Currently, it supports modeling and simulation of Cloud computing environments consisting of both single and inter-networked clouds (federation of clouds). Moreover, it exposes custom interfaces for implementing policies and provisioning techniques for allocation of VMs under inter-networked Cloud Computing scenarios.

III. THE FRAME WORK

A. Overview of Framework:

First, Fig.1 shows the framework based on the layered design of cloud computing architecture The framework also shows the components which are to be managed by the Cloud Security Administration System .The layered designs include :User level, middleware and system level. Middleware can be again classified into user level middleware and core middleware. It is important to note that implementing any of the cloud computing service in the proposed framework will not make the service intrusion-tolerant. The service will be intrusion tolerant only if the protocol or the algorithm upon which the service is based is intrusion tolerant by design.

© 2013, IJCSMC All Rights Reserved

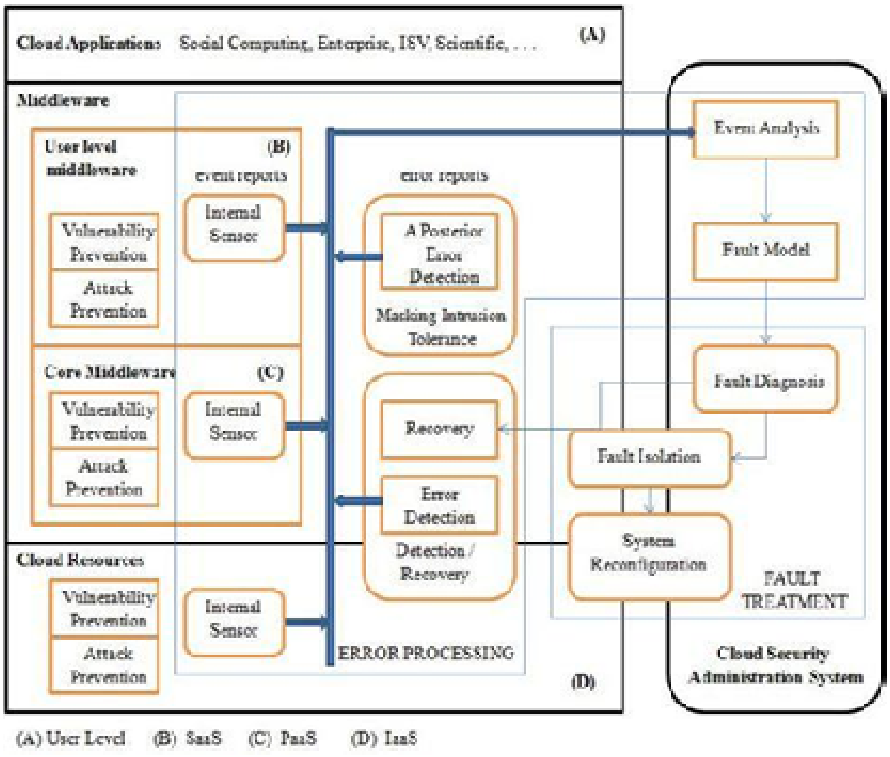


Figure 1: Intrusion Tolerance Framework based on Layered Design of Cloud Computing Architecture

B. Attack and Vulnerability Prevention:

The Attack prevention consists of the introduction of mechanisms such as authentication, authorization and firewalls, which "prevent" attacks in that they "push back" the attacks. With attack taken in the human sense, this includes deterrence measures such as social pressure, laws and their enforcement. Vulnerability includes measures going from semi-formal and formal specification, rigorous design and system management procedures, up to and including user education (e.g., choice of passwords). Attack and Vulnerability Prevention mechanisms must be incorporated in both middleware level and system level.

IV. FRAME WORK VALIDATION

Through this we are representing different techniques for implementing Intrusion detection and tolerance on Cloud computing:

SHIM (System Health and Intrusion Monitoring) is used as an exemplary host-based ID to validate our approach. We formalized all specifications of SHIM which together with a trusted file policy enabled us to reason about the soundness and completeness of the specifications by proving that the specifications satisfy the policy under various assumptions.

DRCP (Dynamic Registration and Configuration Protocol) is an auto configuration protocol in mobile Ad Hoc networks. With respect to this protocol, our approach defines a global security requirement for a network that characterizes the "good" behavior of individual nodes to assure the global property. We formally prove that the local detection rules (identifying activity that is monitored) imply the global security requirement.

OLSR (Optimized Link State Routing) protocol is a proactive, table-driven routing protocol in MANETs. We analyze a specification-based intrusion detection mechanism to detect insider attacks in the OLSR protocol. We proved that the intrusion detection approach, which focuses on monitoring of local behavior, achieves a global integrity of network routing information.

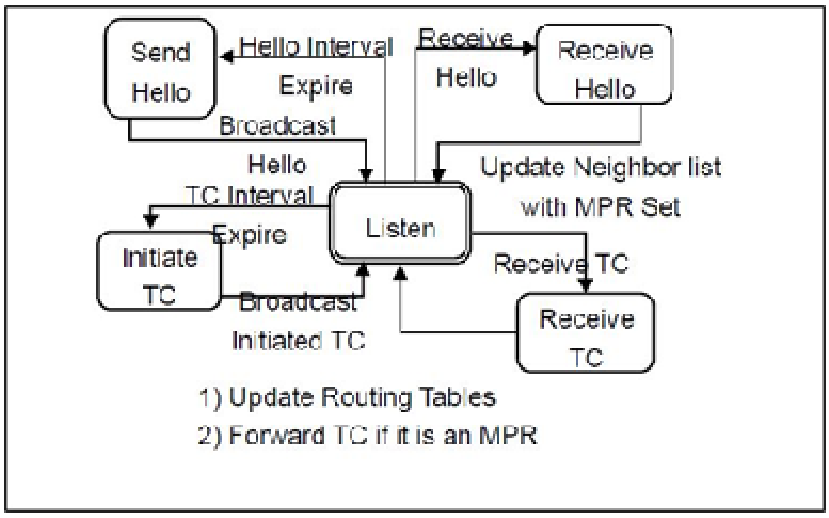
Intrusion detection techniques designed towards wired networks would not be suitable in ad hoc networks due to mobility, lack of fixed infrastructure, etc. For example, a small number of gateways in a wired network can be used to monitor all traffic of the network, but, ad hoc networks typically do not have such choke points. Also, unlike nodes in wired networks, nodes in an ad hoc network may only have limited energy and computation power. Hence, only computationally simple, energy-efficient detection strategies can be used. The detection algorithms must also be distributed as communications with a central computing unit will consume

© 2013, IJCSMC All Rights Reserved

significant energy and bandwidth. These factors motivate the design of a distributed and local specifications focused intrusion detection system geared towards ad hoc networks.

We propose a distributed intrusion detection architecture that allows cooperative detectors to promiscuously monitor all Hello and TC messages by propagating limited control messages. These detectors monitor behavior of their neighbors according to specifications that describe correct behavior of the protocol.

In general, specification-based detection recognizes attacks by comparing the behavior of a system with specifications describing correct behavior of the system. Specification-based detection is particularly suitable for detecting attacks on network protocols because the correct behavior of a protocol is well defined and documented in the protocol specification. The challenge is to extract a suitable correct behavior model from the protocol specification that can be checked at runtime using network monitoring.



A. Simulation Environment:

The feasibility of the proposed framework of Cloud Computing is shown in Figure 2 was simulated using CloudSim toolkit. The computing power in Cloud environments is supplied by a collection of Datacenters that are typically installed with hundreds to thousands of hosts.

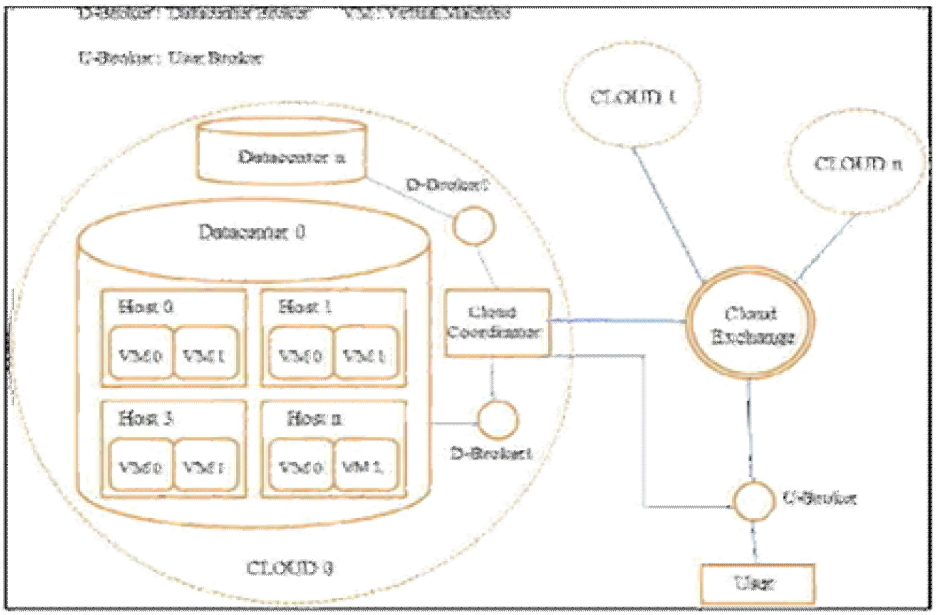


Figure 2: Cloud Computing Simulation Environment

© 2013, IJCSMC All Rights Reserved

B. Design of CloudSim:

a. Datacenter: This class models the core infrastructure level services (hardware) that are offered by Cloud providers (Amazon, Azure, App Engine). It encapsulates a set of compute hosts that can either be homogeneous or heterogeneous with respect to their hardware configurations (memory, cores, capacity, and storage). Furthermore, every Datacenter component instantiates a generalized application provisioning component that implements a set of policies for allocating bandwidth, memory, and storage devices to hosts and VMs.

b. Cloud Coordinator: This component is instantiated by each cloud in the system, whose responsibility is to undertake the following important activities,

i) Exporting Cloud services, both infrastructure and platform-level, to the federation;

ii) Keeping track of load on the Cloud resources (VMs, computing services) and undertaking negotiation with other Cloud providers in the federation for handling the sudden peak in resource demand at local cloud; and

iii) Monitoring the application execution over its lifecycle and overseeing that agreed SLAs are delivered.

c. ***Datacenter-Broker or Cloud-Broker*****:** This class models a broker, which is responsible for mediating negotiations between SaaS and Cloud providers; and such negotiations are driven by QoS requirements. The broker acts on behalf of SaaS providers. It discovers suitable Cloud service providers by querying the Cloud Information Service (CIS) and undertakes on-line negotiations for allocation of resources/services that can meet application's QoS needs. The researchers and system developers must extend this class for evaluating and testing custom brokering policies. The difference between the broker and the Cloud Coordinator is that the former represents the customer (i.e., decisions of this components are made in order to increase user-related performance metrics), while the latter acts on behalf of the data center, i.e., it tries to maximize the overall performance of the data center, without considering needs of specific customers. Cloud brokers are also called as user broker (U-broker) and datacenter broker is D-broker.

d. Host: This class models a physical resource such as a compute or storage server. It encapsulates important information such as the amount of memory and storage, a list and type of processing cores (to represent a multi-core machine), an allocation of policy for sharing the processing power among virtual machines, and policies for provisioning memory and bandwidth to the virtual machines.

e. VM: This class models a virtual machine, which is managed and hosted by a Cloud host component. Every VM component has access to a component that stores the following characteristics related to a VM: accessible memory, processor, storage size, and the VM‟s internal provisioning policy that is extended from an abstract component called the Cloudlet Scheduler.

f. ***Cloud Exchange (CEx):*** It acts as a market maker by bringing together Cloud service (IaaS) and SaaS providers. CEx aggregates the infrastructure demands from the Cloud brokers and evaluates them against the available supply currently published by the Cloud Coordinators.

C. Intrusion Tolerance via Threshold Cryptography (ITTC):

a. Intrusion Detection Module: A Sensor module continuously tests all possible combinations of hosts for valid secret generation. Host(s) present in all the faulty combinations is responsible for generating invalid secret key. Sensor module is capable of detecting intrusions in all „n‟ hosts. This sensor module also generates alert and initiates recovery module when intrusion is detected. Figure 3 shows the algorithm used for detecting faulty hosts when system is being intruded.

© 2013, IJCSMC All Rights Reserved

Algorithm 1 Intrusion Detection Module

Inputs: For (k,n) Shamir's Threshold Scheme
k : Threshold for Share Value
n : Total number of Hosts
OriginalKey : Actual Secret Key **Output:** *FaultyHostList* : List of faulty Host

```
1: procedure
2: for i = 1 TO nCk do
3:    Generate secret key using corresponding k number of shares.
4:    Add generated secret key to NewKeyList
5: end for
6: for i = 1 TO NewKeyList.Length do
7:    if NewKey! = OriginalKey then
8:       Add it to the list of FaultyCombinations
9:    end if
10: end for
11: for i = 1 TO HostList.Length do
12:    if CurrentHost found in each FaultyCombinations then
13:       Add it to the list of FaultyHostList
14:       Alert recovery model for the CurrentHost
15:    end if
16: end for
17: end procedure
```

Figure 3: Intrusion Detection Module

b. Recovery Module: In recovery process, reconfiguration module reallocates all the virtual machines running on the penetrated host(s), and the fault isolation module removes compromised host(s) machine from the Host group constituting a datacenter. Recovery module then invokes key management module for generating and redistributing new secret shares. Figure 4 shows the algorithm used for recovering data center when a faulty host is detected.

Algorithm 2 Recovery Module

Inputs: *FaultyHost* : Corresponding faulty Host

```
1: procedure
2: Reallocate Virtual Machines running on the FaultyHost on other legitimate Hosts
   in the Datacenter
3: Destroy all the Virtual Machines running on the FaultyHost
4: Remove FaultyHost from the Datacenter
5: Invoke Key Management Module to generate new share values for refined set of
   Hosts in the Datacenter
6: Redistribute new secret shares
7: Execute reallocated Cloudlet request in the new setup
8: end procedure
```

Figure 4: Recovery Module

V. CONCLUSION AND FUTURE WORK

In this project, we have designed a framework for intrusion tolerance in Cloud Computing architecture. For the designing a framework, we have used Intrusion Tolerance via Threshold Cryptography mechanism for validation. It is seen that our framework is capable of detecting and recovering intrusions in the Cloud Computing Environment Also, data storage in the cloud can be made intrusion tolerant using proposed framework at the loss of performance. Performance overhead are storage overhead, detection overhead, recovery overhead etc. which increases with the increase in the file size. Future work includes: Refining the implementation of framework components, using proposed framework for other cloud services wherein the algorithm upon which the service is based is intrusion tolerant by design, the framework can be used as a prototype in developing specific intrusion tolerant architectures for different types of distributed applications.

© 2013, IJCSMC All Rights Reserved

REFERENCES

[1] Avizienis, A., Laprie, J.C., Randell, B., Landwehr, C.: Basic Concepts and Taxonomy of Dependable and Secure Computing. IEEE Trans., Dependable and Secure Computing 1(1), 11–33 (2004)
[2] Shamir, A.: How to share a secret. Comm. of the ACM 22, 612–613 (1979)
[3] Saidane, A., Nicomette, V., Deswarte, Y.: The Design of a Generic Intrusion-Tolerant Architecture for Web Servers. IEEE Trans. 6, 45–58 (2009)
[4] Powell, D., Stroud, R.: Malicious-and Accidental-Fault Tolerance for Internet Applications: Conceptual Model and Architecture. Technical Report 03011, Project IST-1999-11583 MAFTIA, Deliverable D21, LAAS-CNRS (January 2003)
[5] Information Technology Infrastructure Library, http://www.itil-officialsite.com/home/
[6] Intrusion Tolerance via Threshold Cryptography,http://crypto.stanford.edu/~dabo/ITTC/
[7] Reynolds, J.C., Just, J., Clough, L., Maglich, R.: On-Line Intrusion Detection and Attack Prevention Using Diversity, Generate-and-Test, and Generalization. In: HICSS 2003, Track -9, vol. 9 (2003)
[8] Pal, P., Schantz, R., Atighetchi, M., Loyall, J.: What Next in Intrusion Tolerance. BBN Technologies, Cambridge
[9] Buyya, R., Ranjan, R., Calheiros, R.N.: Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities. University of Melbourne, Australia (July 2009)

© 2013, IJCSMC All Rights Reserved