Journal of Engineering and Applied Sciences 7 (8-12): 468-482, 2012
ISSN: 1816-949X

An Adaptive Security Framework Delivered as a Service for Cloud Environment

S. Subashini and V. Kavitha
Anna University Tirunelveli, Tirunelveli, Tamil Nadu, India

Abstract: Cloud computing is a flexible, cost effective and proven delivery platform for providing business or consumer IT services over the Internet. Despite of all the hype surrounding the cloud, enterprise customers are still reluctant to deploy their business in the cloud. This study proposes a security framework that provides security as a service for cloud applications. This framework provides security as a single-tier or multi-tier based on the application's demand. Moreover, these tiers are enabled to change dynamically making the entire security system less predictable. The behavior of this framework is designed to be customizable based on the application's importance and is localized. This design will help the cloud in adverse situations of threats because the threats will also be localized and will not pose a threat to the entire cloud. In a cloud where there are heterogeneous asset systems, a single security system would be too costly for certain applications and if there is less security then the vulnerability factor of some applications like financial and military applications will shoot up. This consideration is incorporated within the framework which enables security levels based on the importance of the assets of the services provided by the cloud and their respective clients. The framework was tested and the results were positive.

Key words: Security as a service, cloud computing, adaptive security, multi-tier security framework, India

INTRODUCTION

With the promise of reduced cost, greater uptime, increased flexibility and rapid deployment, cloud computing has piqued interest in the high tech world. Given the claimed benefits, most IT departments are examining whether cloud computing fits their organization's needs and if so how to migrate away from their current on premises infrastructure. The typical concerns of an IT department center on infrastructure availability ease of management, security and cost. The most frequently vetted security concerns focus on the security of the cloud provider itself. While, provider security is a valid concern that must be addressed, the security of programs targeted for deployment in the cloud and the changes that migration to the cloud can introduce are often overlooked. From a technical risk perspective, cloud deployments eliminate existing enterprise network security infrastructure, make defense-in-depth harder to practice because of external dependencies and increase the chance that resources external to the program will be trusted when they should not be. These factors combine to radically change the technical risk exposure for existing enterprise software deployed in the cloud. Moving a program to a cloud provider usually changes the program's threat model in the following areas:

- Communication channels
- Authentication and authorization
- Logging infrastructure
- Data storage
- Encryption
- Environmental dependencies

Cloud computing utilizes three delivery models by which different types of services are delivered to the end user. The three delivery models are the SaaS (Software as a Service), PaaS (Platform as a Service) and IaaS (Infrastructure as a Service) which provide infrastructure resources, application platform and software as services to the consumer. These service models also place a different level of security requirement in the cloud environment. In SaaS, the client has to depend on the provider for proper security measures. The provider must do the work to keep multiple users' from seeing each other's data. So, it becomes difficult to the user to ensure that right security measures are in place and also difficult to get assurance that the application will be available when needed (Choudhary, 2007). Consequently, there is a great deal of discomfort with the lack of control and knowledge of how their data is stored and secured in the SaaS Model. There are strong concerns about data breaches, application vulnerabilities and availability that can lead to financial and legal liabilities.

Corresponding Author: S. Subashini, Anna University Tirunelveli, Tirunelveli, Tamil Nadu, India

According to a recent IDCI survey, 74% of IT executives and CIO's cited security as the top challenge preventing their adoption of the cloud services model (Clavister, 2009). Analysts' estimate that within the next 5 years, the global market for cloud computing will grow to $95 billion and that 12% of the worldwide software market will move to the cloud in that period. To realize this tremendous potential, business must address the privacy questions raised by this new computing model. Cloud computing moves the application software and databases to the large data centers where the management of the data and services are not trustworthy. This unique attribute however, poses many new security challenges (Wang *et al*., 2009). These challenges include but not limited to accessibility vulnerabilities, virtualization vulnerabilities, web application vulnerabilities such as SQL (Structured Query Language) injection and cross site scripting, physical access issues, privacy and control issues arising from third parties having physical control of data, issues related to identity and credential management, issues related to data verification, tampering, integrity, confidentiality, data loss and theft, issues related to authentication of the respondent device or devices and IP spoofing. Cloud requires security which depends and varies with respect to the deployment model that is used, the way by which it is delivered and the character it exhibits. Some of the fundamental security challenges are data storage security, data transmission security, application security and security related to third party resources (Subashini and Kavitha, 2011).

This study is concentrated towards data transmission security and application security and partially towards security related to third party resources. This study proposes a framework that provides security as a service to the services hosted in the cloud and the respective clients of those services. This is a multi-tier security framework (Subashini and Kavitha, 2011) which enables incorporation of localized and customized security module which is dynamic in nature.

Related work: Security in cloud is an emerging topic of research, already addressed in many research and academic publications. A good overview of the issues in cloud is provided by Molnar and Schechter (2010) who investigated the pros and cons of storing and processing data by the public cloud provider with regards to security. They detail about the new forms of technological organizational and jurisdictional threats resulting from the usage of cloud as they also provide a selection of counter measures.

The different threat and attack models given by Akhawe *et al.* (2010) can be used to formally analyze the attacks in cloud computing scenarios. However, their approach is limited to HTTP communication only. The model does not take into account application layer messages.

Jung and Chung (2010) proposed an Adaptive Security Management Model for cloud computing algorithm. They suggest an adaptive access algorithm to decide the access control to the resources using an improved RBAC Technique. The proposed model determines dynamically security level and access control for the resources. But this model is based on provision of security based on cloud providers' decision and mainly considers different types of resources to arrive at the security level and access control. The model is targeted towards decisions of the client and services along with the resources to arrive at security levels. Also, the model is framed considering the cloud provider also as an third party untrusted provider, thus making the system non-vulnerable even at the hands of the provider.

Gruschka and Iacono (2009) showed how XML signature wrapping attacks can be performed to attack Amazon's EC2 service. They detailed a vulnerability that enabled an attacker to execute operation on the cloud control while having possession of a signed control message from a legitimate user.

Yunis (2009) outlines six security considerations for cloud computing namely resource sharing, data ownership, reduced encryption in favor of speed, refusal of services, data loss due to technical failure and attackers going after provider or the implementation. He also proposes a theoretical model for overcoming these issues through management of policies. For example, he proposes to classify the policies based on different types of data like client financial data, intellectual property and so on. But creation and management of these policies are practically cumbersome and inefficient. Though, many of the security issues in the past were due to inefficient policies, enabling an efficient policy is next to impossible. Policies can only be an additional measure but as long as the security framework is not efficient even the most strategically created security policy will fail.

Though, existing cloud providers use APIs that have the structure of web services standards such as SOAP and also standard cryptographic primitives for authentication is done through SSL protocol. Also, they have IT infrastructure comprising of proxies and gateways containing malware and intrusion detection techniques. There are two problems arising out of this. The first one is that the API structures are still proprietary because they use the provider's own semantics within the

standard structures. This will have a great impact on user's ability to move their data from one provider to the other. The other one is that the public key cryptography, a central concept in cryptography is used to protect web transactions and its security relies on the hardness of certain number theoretic problems. These are also the main place where quantum computers have shown to have exponential speedups. These problems include factoring and discrete log, computing the unit group and class group of a number field (Gentry and Szydlo, 2002) and Pell's equation (Gentry *et al*., 2008). The existence of these algorithms implies that a quantum computer could break Diffie-Hellman and elliptic curve cryptography (Paryasto *et al*., 2009), RSA which are currently used as well as potentially more secure systems such as the Buchmann Williams key exchange protocol. It's just a matter of processing power that is required to crack these cryptic techniques. In that case, employment of standard cryptographic techniques will be efficient only over a matter of time. As processing power gets exponential and quantum algorithms like Simon (1997)'s quantum algorithm gain better strength, these cryptographic techniques are bound to be disrupted may not be immediate but in near future.

Bertino (2004) proposed a model for secure publication of XML documents. In the presence of third party publishers, the owner of a document specifies access control policies for the subjects. The subjects obtain the policies from the owner when they subscribe to a document. When the subject requests a document, the publisher will apply the policies and give fragments of the documents to the subject. Now, since the publisher is untrusted, it may give false information to the subject. Therefore, the owner encrypts various combinations of documents and policies. Using Merkle signature and the encryption technique the subject verifies the authenticity and completeness of the document. This model provided holds good for conventional scenarios but in case of cloud systems the range of vulnerabilities are even higher and hence should be targeted in a different way. The model proposed by Bertino (2004) can still be used in conjunction with the framework proposed in the study. The framework proposed in the study enables to provide confidentiality behind this level. This refers to the requirement of a subject in the cloud receiving a response to an access request must be able to verify the completeness of security of the response. This can pertain to data or any document. The framework proposed in this study provides a solution for this in the path of using checksum validations for every request and response, the details of which will be discussed in the this study.

One of the recently exploited vulnerability is the cloudburst exploitation of vulnerability in VMware display functions in order to execute code from within a guest VM into the controlling host. Once exploited, the exploit tunnels a connection over the frame buffer of the guest to communicate with the host (http://immunityinc.com/news-latest.shtml). The framework provides a different solution to tackle this situation which is irrespective of the policies being executed in the guest or host VM's.

SECURITY AS A SERVICE

According to the basic concept of cloud computing, anything and everything should be delivered as a service. Similarly, security can also be provided as a service. This security service can be delivered based on demand from software services or from their corresponding clients. This service need not be a single service. A group of security related services can combine together to deliver security as a service. Though, there are threats regarding security while moving enterprise applications and data to the cloud if the entire service framework is designed properly, the cloud environment can provide a better security to the enterprise applications and data in contrast to the security framework available within the enterprise. This is because of the fact that cloud, by its nature has better and higher processing capabilities to provide security functionalities. This service should not only be delivering cryptographic and firewall functionalities. Instead its scope is wider with concerns regarding the following:

- Authentication and authorization
- Data exchange within the cloud
- Requirement based security
- Localization of security procedure
- Intrusion detection

In case of a public cloud, the robustness of this security service is of utmost importance. This will be the service which would enable trust within the cloud users to use the services of the cloud and to exchange data within the cloud environment. The cloud providers can use the robustness of the module to attract users towards their environment and create a trusty environment for enterprise owners to move their data and business into the cloud. Though, there are many concerns for providing such a service within the cloud, this study provides a overall robust framework for providing security as a service within the cloud. This framework can be fine tuned and used in a practical cloud environment with some add-ons. This framework and appropriate results are discussed in this study.

MULTI-TIER SECURITY FRAMEWORK

This framework provides security as a service to other services and clients in the cloud environment. The security service is designed to comprise of three inter related modules namely the functional security module, transaction security module and the End Point (EP) module. Each of these modules in itself is made up of collection of services. Each one of these modules provides certain functionalities independently and other functionalities as a group. Figure 1 shows the elements of this framework. The functional security module is the one which is responsible for providing services related to encryption, decryption, security agents, firewall restrictions and validations.

The transaction security module delivers services to secure applications and the data that is flowing within the cloud environment. This module contains the elements which forms single or multiple tiers of security systems as required for the transaction. The Tier Elements (TE) are also services catering to the different levels of security which will be discussed in detail in the later part of the study. These tier elements can communicate with the functional module for getting security functionalities delivered. The end point module is a series of service end points which encapsulates the services of the other two modules. The user (client or service) communicates with these end points for initiating a communication with other services. The user in addition to this can demand various levels of security required for the transaction. The sequence of working of the framework is as follows: Fig. 2 explains steps 1 through 9.

Client authenticates into the cloud environment. Client contacts the SR (Service Registry) for discovering software services and gets their corresponding end point details. Client requests the SR for end point of security services and SR acknowledges with a possible SSE (security service end point for example EP3 as shown in Fig. 1). Client contacts the SSE and sends the end point details of the software service and requests for initiating

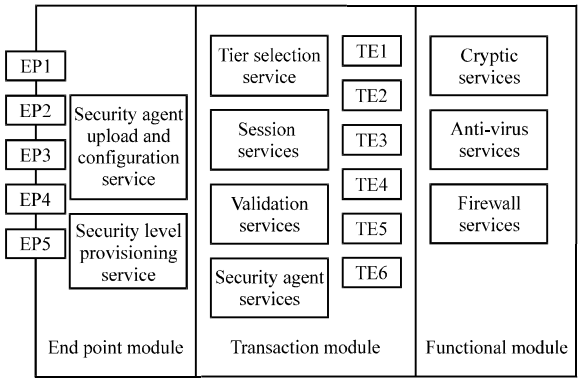


Fig. 1: Security framework

a communication with that particular service. SSE requests the client for the security levels required for the transaction. This is done by the security level provisioning service of the end point module. The different levels of transaction are a list provided by the security service. For example, it can range from Level 1 (L1) to 5 with 1 being the lowest security level and 5 being the highest. Possible levels of security are discussed later stage. Client can select the security levels based on the importance of the asset decided by the client. The security service can be of pay per use model and the cost of usage depends on the level of security required by the user. The security service can also provide some basic security (for example, L1) as being free of cost.

SSE communicates with the software service end point and requests for the level of security demanded by the service for the transaction. There can be a situation where client does not demand security and the service demands security and it can be vice versa. In such a scenario, the side which is not interested in security is forced to conform to the security levels of the other side. But, the pay per use model of the security service should not incur cost to the side which is not interested in the security. Here, there arises another issue. There may be different transaction going on with the service and different clients may require different levels of security. Suppose client 1 requires normal security and client 2 requires high security and the service by itself needs only medium level security. In that case, the service level provisioning service makes sure that all transactions between services and any of the clients are at the high level. So at any given point of time, all transactions with a service will be at the highest level which is requested by even one of the participating clients. If even one client requires more security than the other clients then all the clients are shifted to higher security mode transactions. This makes sure that there is no vulnerability because of the difference in security levels between participating resources. The levels can be subdued when the client requiring higher security is out of the access with the service. In a practical scenario where client 1 and 2 are accessing a service from a virtualized guest OS of a host if client 1 is communicating with the service with higher security levels and client 2 is communicating with lower security levels then there is a possibility that client 2 is vulnerable. In order to avoid this, all the clients are transferred to higher levels of security at runtime. The security levels are dropped and enhanced dynamically based on the resources participating in the transactions. This methodology will help in solving problems like the cloudburst intrusions.

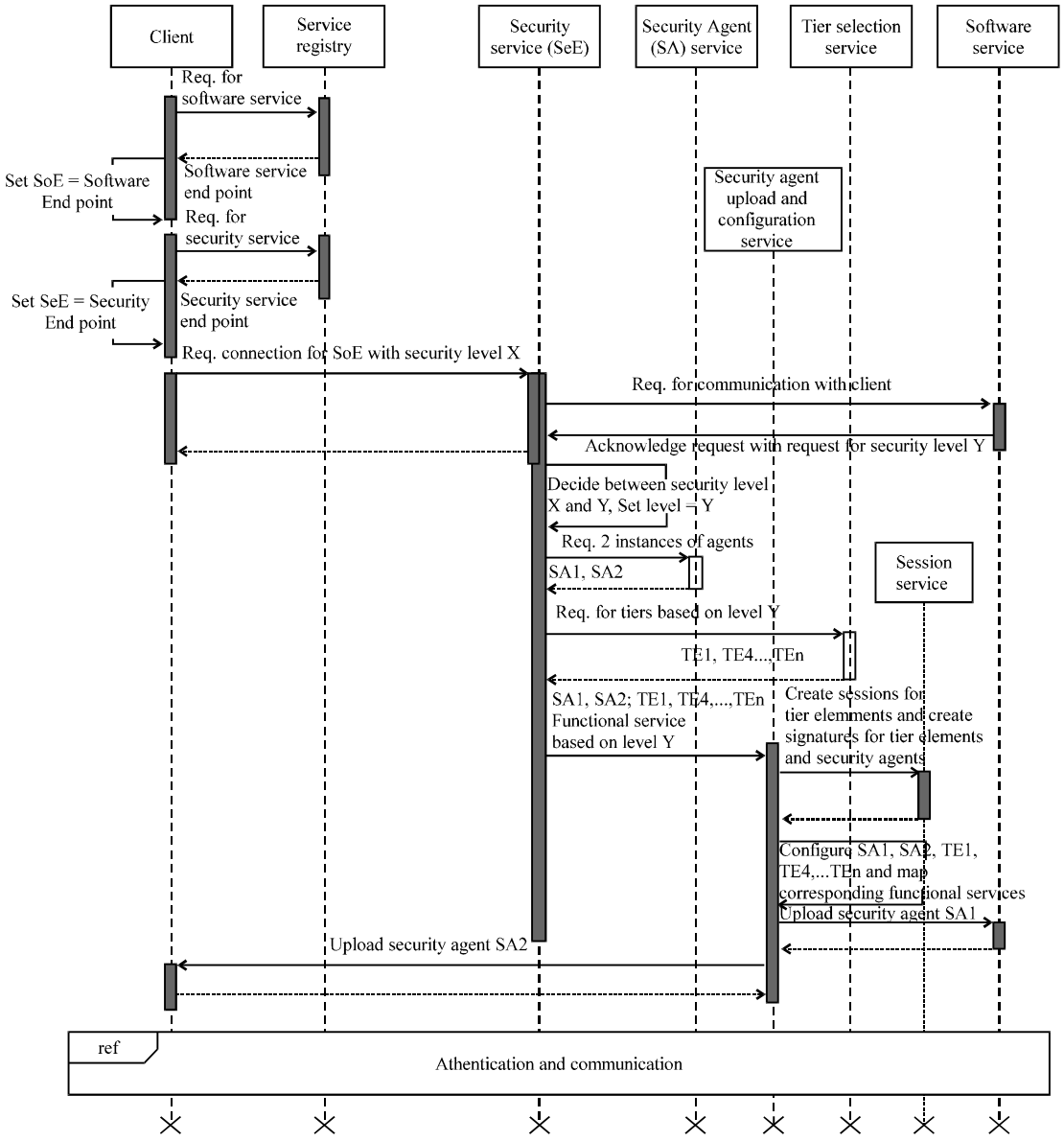


Fig. 2: Establishing connection between client and service based on security demand

SSE communicates with its functional security module and forces the client and service to use an instance of a Security Agent (SA). This is done by the security agent upload and configuration service of the end point module in cooperation with the security agent service of the transaction module. This security agent is dynamically configured to cater to the security levels of the transaction and is dedicated to the transaction. The security agents can either be downloaded to the client and service side or can be in the network at the security service side. Locally downloaded agents will provide better security and can be devoid of malware attacks and attacks like IP spoofing. SSE selects tier elements (for example, TE1 and TE4) using the tier selection service of the transaction security module and configures them within the security agents. Session service of the transaction module creates new sessions within the tiers for this particular transaction and session based signatures are provided for each of these tiers. The security agents at both sides are provided with encrypted unique signatures and this signature is exchanged between the client and service for each communication to ensure integrity of data. The signature of the security agents are registered within the tiers in a dedicated

Table 1: Security levels

Security level	No. of security tiers	No. of functional security services	No. of private security services
1	1	All basic services	None
2	3-5	Basic services + up to 5 pre-defined functional security services (1 service per tier)	None
3	3-7	Basic services + up to 10 pre-defined functional security services (2 service per tier)	None
4	3-10	Basic services + up to 10 pre-defined functional security services (2 service per tier)	Up to 3 private services (can be coupled with functional services)
5	3 to unlimited tiers (pay per tier per usage)	Basic + any functional service (3 services per tier)	Unlimited private services (pay per private service per coupled usage with tiers)

session. The number of tier elements and nature of the tier elements are decided by the security levels chosen by the users. This is also based on other client participating in transaction with this service. The nature of the security levels is decided by the cloud provider and it can be allowed for minimal customization from the user side. A possible (not compulsory) security level definition is shown in Table 1. For example, level 3 of security service can have the possibility between 3-7 tiers and the user can be allowed to customize this number. Or another method of providing customization is by allowing the user to map the transaction and functional services. For example level 2 of security can be of 2 tiers from the transaction services and three functions (firewall, anti- virus service and cryptic service) from the functionality module of the security service. The user can customize the 1st tier as anti-virus service and the 2nd tier as a cryptic service. The order of functionality services associated with the tiers can pose some issues. For example, if the 1st tier is cryptic service and the 2nd tier is anti-virus service then the anti-virus service is of no use because it cannot read the encrypted data. Such types of clashes in sequences are taken care by SSE. It either does not allow the user to choose such a sequence or it overrides the sequence at runtime. So, the client service communication will be on a 2-tier security system with enabling of anti-virus and cryptic service. Let us consider this example itself for explaining the fore-coming steps. The security agents are configured by the SSE for this multi-tier security system that has to be provided for the communication between the client and the application delivered by the software service. Figure 3 shows steps 10 through 16.

All data that is flowing between the client and the service passes through this multi-tier security system. This involves data right from the stage of authentication and till the communication is closed. Whenever data is obtained at either side, it is checked for clearance from the multiple tiers. If data is not authorized by all the tiers then the data is rejected and re-configuration of security agents is done before continuation of further communication.

Let us consider a one cycle of communication between the client and the service in this scenario. Client sends data to the security agent (either residing within the client or in a networked security system). Security agent adds its private key (here it is a encrypted signature and this can be modified to any other robust security technique based on requirement) to the data and sends the data to the 1st tier configured within the agent. Along with this data, the security agent calculates the checksum of the data (in an encrypted format) along with its signature. This checksum is added because at each stage the checksum is checked with the data to avoid code or data from malicious user to be attached with the data and sent to the service. This enables that the response to a request is just the actual intended response and is not containing any malicious data. This can be used along side the methodology proposed by Bertino (2004).

As per example, the 1st tier gets the data and checks for the signature of the security agent against the one already registered for the session. It also verifies the checksum value and the actual checksum of the data. Once it is validated, this tier element communicates the functional services and does an anti-virus check over the data and then adds the dedicated session based tier signature to the data and also the checksum. This addition of checksum again in this level is because that there may be situations where the functional services change the data. For example when cryptic services are involved, the checksum of the data after encryption will be different. So at every tier, the tier element calculates the checksum and adds it to the data along with its signature in an encrypted way.

The 1st tier service then communicates with the security agent at the client side and gets the end point information of the 2nd tier and then sends the data to the 2nd tier. The 2nd tier validates the signature of the security agent and then retrieves signature information of the 1st tier from the security agent and then validates the same with the signature present in the received data and also the checksum of data. Once the validation is done, the 2nd tier takes the help of the functional services to provide cryptic services to the data (maybe encryption of data). The 2nd tier element adds its corresponding signature and checksum of data and sends the data to the software service.

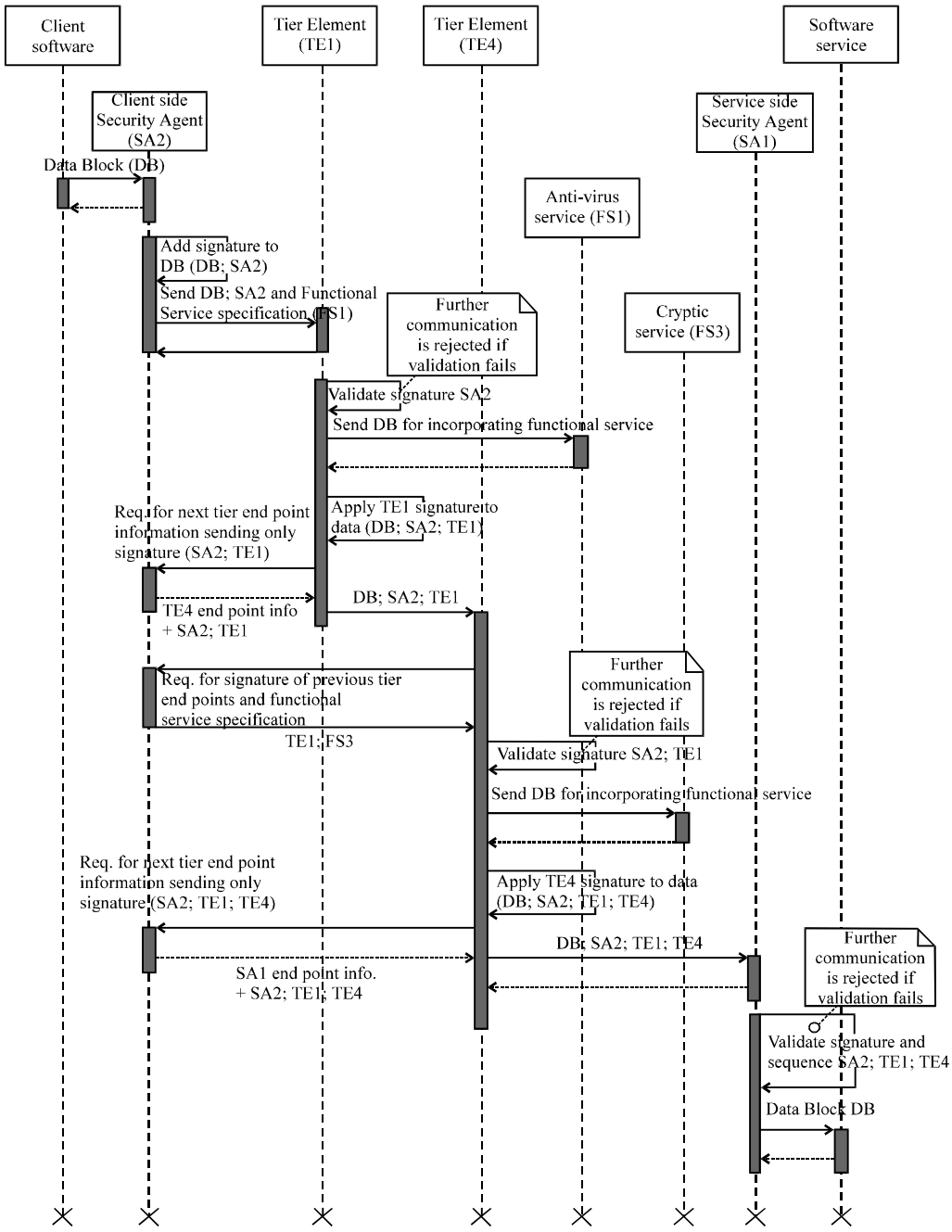


Fig. 3: Communication cycle between client and service using the multi-tier framework (based on the example discussed)

The security agent at the software service side receives the data and validates for the signature of the client side security agent, signature of the 1st and the 2nd tier and then adds its signature and sends the data to the 2nd tier again for decryption of service (as configured in the security agent, an improvised procedure is discussed in this study). The 2nd tier validates all the signatures once again along with the signature of the service side security agent and then provides decryption services using the functional modules and provides the data to the actual software service.

The proposed sequence can be a little predictable because the tiers remain constant throughout the communication unless there are any new resources requiring higher level security which will be automatically elevated for client by SSE. A predictable system is target

for hackers for a possible attack. Though, different cloud systems present today claim that they use standard methodologies for security, they customize the standard methodologies so that any hacker does not predict the system easily. Using a common model for security which is also predictable by any hacker and then applying cumbersome policies to avoid the hacker to access the system had always been inefficient. Always there are vulnerabilities and loop holes that were identified in many environments (Molnar and Schechter, 2010; Akhawe *et al*., 2010; Yunis, 2009). So, this methodology can be fine tuned to make it more unpredictable. Such a mechanism is discussed in this study.

DYNAMIC TIERS

Dynamism in this framework can be brought in by a couple of ways, one by changing the tier elements over the communication period and the other way is by deciding the association of the tier and the functional services during runtime instead of configuring the security agent.

In the 1st case, the tiers can be changed during runtime at a random phenomenon. This task should be taken care of the session services. At a random time frame, this service in interaction with other required services in the transaction and end point module can decide a new set of tiers that should be involved in further communication. Correspondingly, the security agents in the client and service side should be reconfigured and new sessions and signatures should be created in the security agents and the tiers and exchanged between each other and further communication can proceed. It might seem to be a costly procedure but keep in mind that the entire framework is running on a cloud which by default has high processing power. And it is not costly as the functionality is a security provision for a demanding asset.

In the 2nd case, every tier is not mapped with any functional service at the starting of the communication. Instead the mapping is done at runtime. The list can either be managed by the security agent or by the security service itself. The list contains the list of tiers under consideration and the list of functional services but the tiers and functional services are not mapped. For every communication transaction, a possible mapping is done at runtime and executed. For example, consider a 2-tier system (say TE2 and TE5) with two functional services (functional service 1 and functional service 2). In the first communication cycle, TE2 is mapped to functional service 1 and TE5 is mapped to functional service 2 and in the subsequent communication cycle TE2 caters functional service 2 and TE5 provides functional service 1. This mapping can be managed by the agent itself or by the security service. This scenario cannot be followed if the composition of functional services is conflicting. For example if the functional services are anti-virus service and cryptic service then these services cannot be inter changed. This should be taken care by the security service.

A combination of the 2 cases would be a very effective one. That is by doing a runtime mapping of tiers and functions and also changing the tiers during runtime by reconfiguring. This methodology is still costlier but if security requirements demand such a scenario, it can be provided based on demand. This can also be included as higher levels of security (say, level 6 as a combination of any level between level 1-4 and dynamic tier changes, level 7 as a combination of any level between level 1-4 and runtime mapping and level 8 as combination of level 6 and 7) and can be deployed purely based on demand by the users.

DEDICATED AND PRIVATE SECURITY

This proposed framework is based on the conceptualization of the cloud security based on real world security system where in security depends on the requirement and asset value of an individual or organization. For example, a normal human does not require personal security but a well known personality needs a body guard, an organization needs a set of security persons and a state or country have their mass military to safe guard their assets. The intense of security is directly proportional to the value of the asset it guards. In a cloud where there are heterogeneous systems having a variation in their asset value, a single security system would be too costly for certain applications and if there is less security then the vulnerability factor of some applications like financial and military applications will shoot up. On the other side, if the cloud has a common security methodology in place, it will be a high value asset target for hackers because of the fact that hacking the security system will make the entire cloud vulnerable to attack. In such a scenario if customized security is provided as a service to applications, it would make sense.

The methodology discussed in the previous studies can cater to the security needs of normal clients and enterprise business. But for high value financial units and other assets of the government, critical military applications and sensitive political data, the security needs are still high. In for such scenario, the cloud can provision customizable add-ons to the security service. In fact, this would turn out to be costly but in comparison to the value of the assets under consideration, it would turn out to be cheap and robust. This can be done by two ways, one by dedicated security where in the security service provides tiers and functional services which are dedicated only for that particular application. It means

that though the concept of session creation within the tier holds good even in this case yet sessions for other user's communication is not allowed to be created within these tiers. The tiers are completely dedicated for those security critical applications and though the processing power of the tier is more, it is not used taking security concerns under consideration.

The second way is by allowing the applications to use their own private security services in addition to the functional services provided by the cloud (Level 4 and 5 of Table 1). In this case, some tiers are associated to the private security services of the client or the software service. This is in addition to the association with the functional services. This methodology will provide trust enabling within the cloud users. These high value applications can be hosted within a public cloud and the enterprise can deploy a private cloud just for provision of private proprietary security service. So, the application is deployed in the public cloud and uses all the functionality of the different services hosted in the cloud including the security services of the cloud. In addition to that the private security services are used. The combination of the above two methods provide a mightier security shield for the application but if the application demands this can also be deployed. A combination of dedicated security and one of the security levels within level 1-4 can be considered as level 9 of security service. A combination of level 9 and 8 can be considered as level 10 which will the highest level of security service provided by the cloud. A smart decision is required while selection of security levels for the applications and their corresponding communication with clients. In this scenario, though cloud takes care of providing the security, a part of the decision making is given to the client which enables trust within the user and it is also unpredictable by an attacker.

SECURITY OF CLIENT VERSUS SERVICE

When it comes to communication, it is always between two parties, the client and the service. The levels of security demanded by the client might be different than the level of security demanded by the service. The responsibility of the security service is to cater to the demands of both the parties. The decision making is designed within the security service so that when two parties demand for different levels of security, the service should deploy a security methodology for the communication that is more secure out of the two. Otherwise, it can be a combination of the security levels of both the users. For example if a client wants to communicate with software service at level 1 security and if the service demands for level 3 security service then the security service should deploy level 3 security on both sides so that the communication is secure between the clients and services thereby catering to the needs of both the users.

Also most importantly if there is any other client involved in transaction and is requiring an higher level of security then the transactions between all the clients and the service is elevated to the higher level as required by the new client. This consideration need not be taken if the new client used dedicated or private security because there will not be sharing of partitioned physical resources (like scenarios in VM). In case of usage of third party services, the client can demand a higher level of security to cater for its communication with the third party service. This can up to some extent cater to the problems faced with security concerns with third party services.

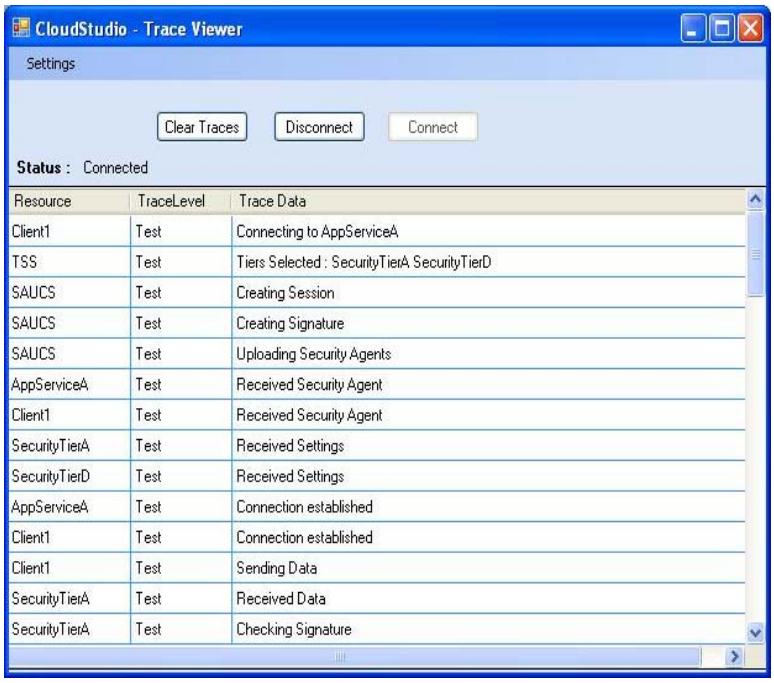
IMPLEMENTATIONS AND RESULTS

The proposed framework was implemented and deployed in Eucalyptus 1.5 cloud environment and the details of which are as follows: The Eucalyptus cloud environment was deployed in 6 Intel core i5 machines running on Ubuntu. Windows 2003 server images were hosted on three of these machines. Windows XP images were hosted on three of these machines. The services required for testing the framework were implemented and deployed in these machines. The details of the deployment are given in Table 2.

Table 2: Test setup

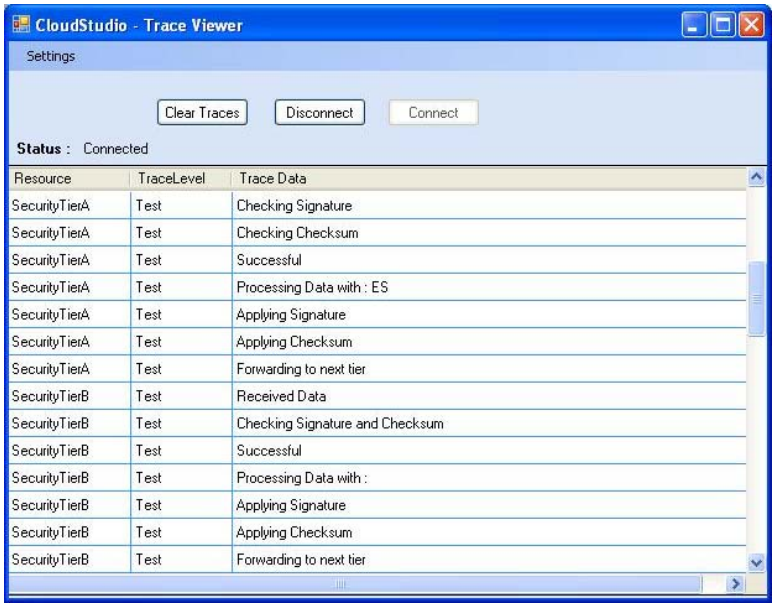
Machine	Host OS	Guest OS	Services
1	Ubuntu	Windows 2003 Server	Security Agent Upload and Configuration Service (SAUCS), Service Registry service (SR) and Trace Service (TS)
		Windows 2003 Server	Security level Provisioning Service (SPS)
2	Ubuntu	Windows 2003 Server	Tier Selection Service (TSS) and Session Services (SS)
		Windows 2003 Server	Security Agent Service (SAS) and Validation Service (VS)
3	Ubuntu	Windows 2003 Server	Security tier A and Security tier B
		Windows 2003 Server	Security tier C and Security tier D
4	Ubuntu	Windows XP	Security Tier E and Security tier F
		Windows XP	Encryption service and anti-virus service
		Windows XP	Customized functional service (CS)
5	Ubuntu	Windows XP	AppService A, AppService B
		Windows XP	AppService C
6	Ubuntu	Windows XP	Client 1 and 2
		Windows XP	Client 3 and 4

Initially, the framework was tested for happy path scenarios. A trace service was written to analyze and test the environment. A separate test level traces were written in different parts of the service and clients while implementing. Initially one client (Client 1) was connected to the AppServiceA with 2 tiers of security. Data was transmitted from the client to the service in intervals keeping the connection alive. During this process, another client (Client 3) residing in a different guest OS of the same machine was triggered to establish connection and transmit data to the same service with 3 tiers of security. The trace for this scenario is shown in Fig. 4-7. Figure 6 and 7 shows the scenario when the 2nd client starts its interaction. The Tier Selection Service (TSS)



Resource	TraceLevel	Trace Data
Client1	Test	Connecting to AppServiceA
TSS	Test	Tiers Selected : SecurityTierA SecurityTierD
SAUCS	Test	Creating Session
SAUCS	Test	Creating Signature
SAUCS	Test	Uploading Security Agents
AppServiceA	Test	Received Security Agent
Client1	Test	Received Security Agent
SecurityTierA	Test	Received Settings
SecurityTierD	Test	Received Settings
AppServiceA	Test	Connection established
Client1	Test	Connection established
Client1	Test	Sending Data
SecurityTierA	Test	Received Data
SecurityTierA	Test	Checking Signature

Fig. 4: Happy path traces 1



Resource	TraceLevel	Trace Data
SecurityTierA	Test	Checking Signature
SecurityTierA	Test	Checking Checksum
SecurityTierA	Test	Successful
SecurityTierA	Test	Processing Data with : ES
SecurityTierA	Test	Applying Signature
SecurityTierA	Test	Applying Checksum
SecurityTierA	Test	Forwarding to next tier
SecurityTierB	Test	Received Data
SecurityTierB	Test	Checking Signature and Checksum
SecurityTierB	Test	Successful
SecurityTierB	Test	Processing Data with :
SecurityTierB	Test	Applying Signature
SecurityTierB	Test	Applying Checksum
SecurityTierB	Test	Forwarding to next tier

Fig. 5: Happy path traces 2

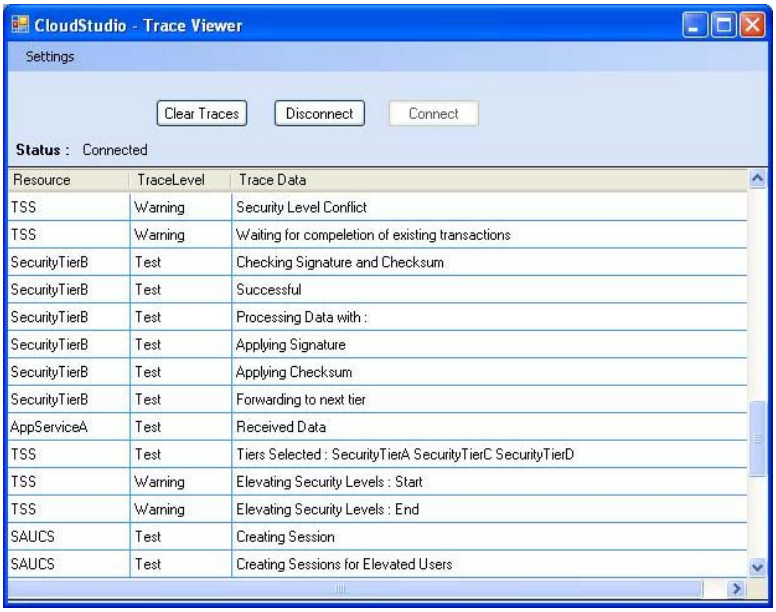


Fig. 6: Happy path traces 3

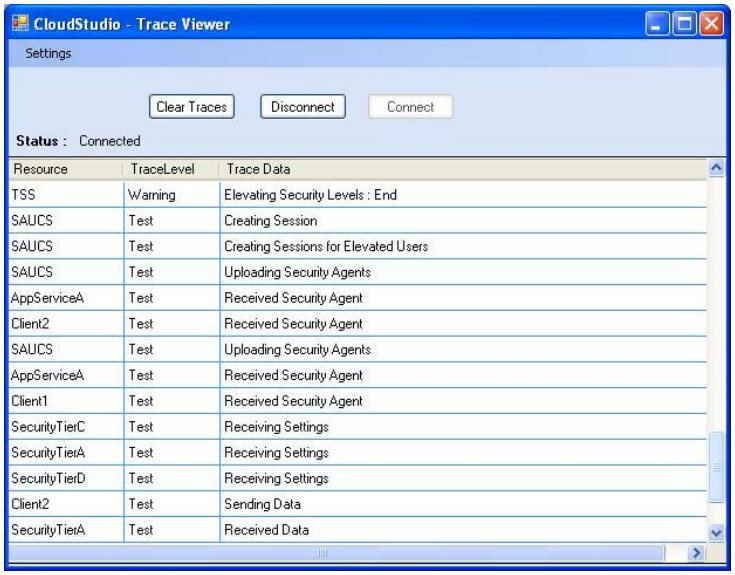


Fig. 7: Happy path traces 4

figures out the change in requirement of security level and elevates the security level of all the existing clients. The tests were further enhanced by increasing the number of tiers required for each client. There were no issues encountered during these tests except for slight delay in the receiving of the data. The performance of the framework with increase in the number of tiers were studied by synchronizing the time between the clients and the services and sending certain amount of data from client to service and calculating the time taken. The results of the performance test are listed in Table 3.

Table 3: Performance data

No. of tiers	Datablock size (kb)	No. of datablocks	Time taken (m sec)
0	200	10	42
	500	20	239
1	200	10	47
	500	20	247
2	200	10	48
	500	20	251
3	200	10	53
	500	20	267
4	200	10	52
	500	20	273
5	200	10	61
	500	20	277
6	200	10	66
	500	20	283

Though, there is a delay in data transmission with the increase in the number of tiers, the delay is only meager compared to the elevation in the security level. In a real cloud environment, these performance data will be much better because of better processing power and high speed network connections. Next, the rainy path scenarios were tested. The rainy path scenarios were created using Back Track Linux (http://www.backtrack-linux.org) and Metasploit (http://www.metasploit.com). Vulnerabilities were manually created in some of the machines running the services and the vulnerabilities were checked and validated using Matasploit before starting of testing the framework. Different attacks (wrapping attack, scripting attack) were hosted using Matasploit and exploitation of the transactions was tried out. Figure 8-10 show the snapshots of the external triggers done for creating attacks.

When the attacks were launched with 0 tiers and without encryption, all the attacks were completely successful except for some failures in the machine which had Host Intrusion Prevention (HIP) system. When encryption was enabled, the attacks were successful in most of the cases but the data obtained using the attack was not useful. When the tiers were included, the intrusion was successful but further communication was denied because the tiers detected the intrusion and recreated sessions and tier sequences. Though, intrusion was possible, only fragments of encrypted data were

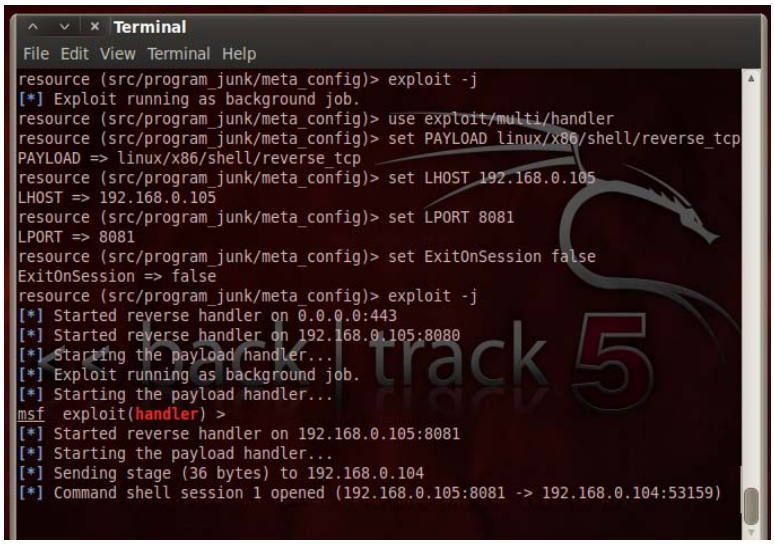


Fig. 8: Metasploit attacks-1

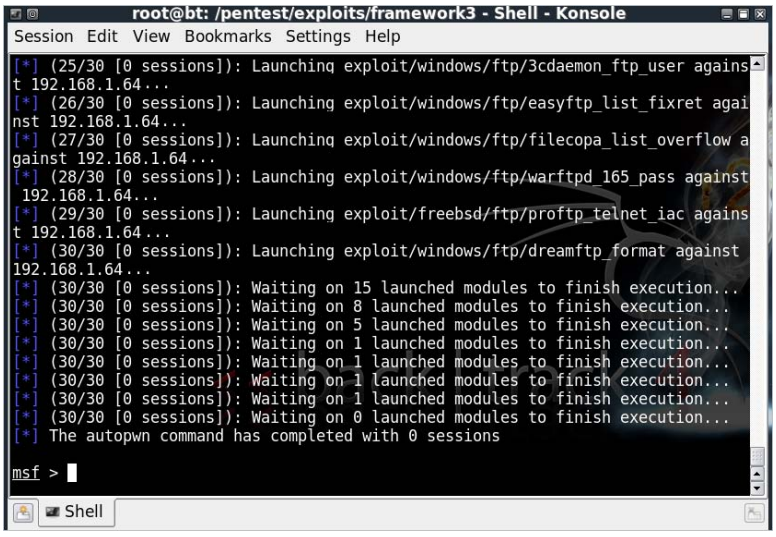


Fig. 9: Metasploit attacks 2

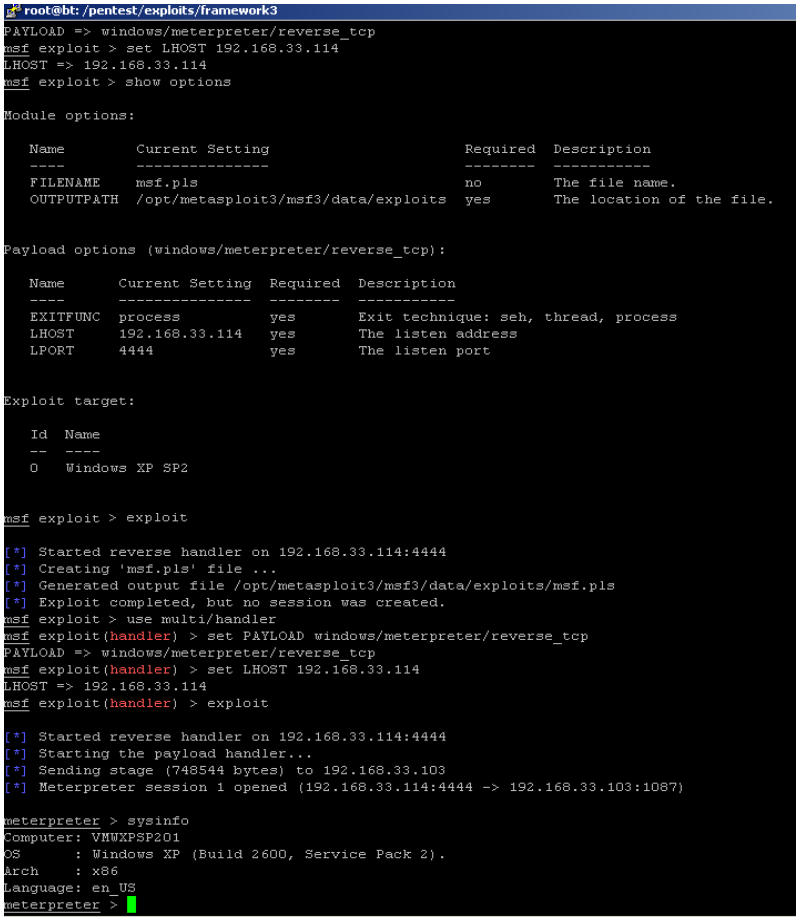


Fig. 10: Metasploit attacks 3

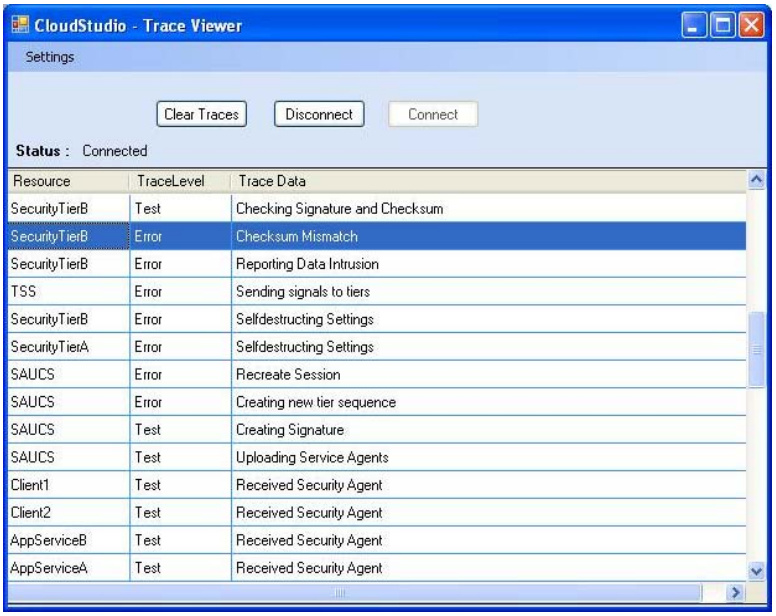


Fig. 11: Intrusion detection

obtained and further attacks were interrupted. Thus, none of the attacks were successful. Additionally when dynamic changes of tiers were enabled over random time logic, the intrusion was possible but no data was obtained in any of the attacks. When data was changed and sent by the attack, the tiers identified a mismatch in the checksum of the data and recreated sessions and tier sequences. The snapshot of the trace which shows the detection of intrusion and recreation of session and tier sequence is shown in Fig. 11.

The analysis of the results thus shows that the proposed framework is robust and suitable for being enabled as a security framework for cloud environment. The performance results also show that there is only a meager change in the performance but if deployed in a real time cloud environment which has higher capabilities, these results would be better and as far as the nature of the current cloud environments are concerned, it has high processing capabilities but lacks in security model. Thus, trading off a bit of performance for the enablement of efficient security holds good.

CONCLUSION

The multi-tier framework discussed in this paper is a robust framework that can be deployed within a cloud to cater to the needs of clients demanding diverse security requirements. This is an overall framework and is not constrained to any security techniques. Any existing security technique and upcoming inventions of security techniques can be accommodated as services within the functional module of the framework and thus makes the framework more flexible. Even a private security technique can be deployed as a service and can be used along with this framework which enables enterprise users to throw away their fears about the security of the cloud and will step into deploying their applications in the cloud utilizing the full power of the cloud and yet not compromising the security concerns. The unpredictable nature of the security framework will motivate the users and build trust over the cloud. And adding to it is the flexibility they get to plug in their private security module along with the security functionalities provided by the cloud. Security will be provided as a service which users can use based on their need and the decision can be done by the user for the cost they need to pay for the security in trade off to the value of the assets under consideration. This framework allows a normal user who does not need any security to use the same cloud in which high value assets are flowing around. In fact, the normal user need not pay anything for the security but the high value assets need to trade off and spare out reasonable pennies for enabling security. Thus, this framework provides flexibility in security services for different segments of the cloud environment. This framework can enable robust security for data flowing within the cloud; applications provided as service within the cloud and to some extent provide security for interaction with third party services. The future research will be deploying the framework in a real time public cloud and testing it.

REFERENCES

Akhawe, D., A. Barth, P.E. Lam, J.C. Mitchell and D. Song, 2010. Towards a formal foundation of web security. Proceeding of the 2010 23rd IEEE Computer Society Foundation Symposium, May, 2010, IEEE Computer Society Washington, DC, USA., pp: 290-304.

Bertino, E., 2004. Selective and authentic third party distribution of XML documents. IEEE Transactions Knowledge Data Eng., 16: 1263-1278.

Choudhary, V., 2007. Software as a Service: Implications for Investment in Software Development. Proceedings of the 40th Annual Hawaii International Conference on System Sciences, January 3-6, 2007, Big Island, Hawaii.

Clavister, 2009. Security in the cloud, clavister white paper. IDC Enterprise Panel. http://www.clavister.com/documents/resources/white-papers/clavister-whp-security-in-the-cloud-gb.pdf.

Gentry, C. and M. Szydlo, 2002. Cryptanalysis of the revised NTRU signature scheme. Proceedings of the Eurocrypt'02 International Confrence on the Theory and Applications of Cryptographic Techniques: Advance in Cryptology, April 28-May 2, 2002, Springer-Verlag, London, pp: 299-320.

Gentry, C., C. Peikert and V. Vaikuntanathan, 2008. Trapdoors for hard lattices and new cryptographic constructions. Proceeding of the 40th ACM Symposium on Theory of Computing, May 17-20, 2008, New York, USA., pp: 197-206.

Gruschka, N. and L.L. Iacono, 2009. Vulnerable cloud: SOAP security revisited. Proceedings of the IEEE International Conference on Web Services, July 6-10, 2009, IEEE Computer Society, Sankt Augustin, Germany, pp: 625-631.

Jung, Y. and M. Chung, 2010. Adaptive security management model in cloud computing environment. Proceeding of the 12th International Conference on Advanced Communication Technology, February 7-10, 2010, Busan, South Korea, pp: 1664-1669.

Molnar, D. and S. Schechter, 2010. Self hosting vs. cloud hosting: Accounting for the security impact of hosting in the cloud. Proceedings of the 9th Workshop on the Economics of Information Security, June 7-8, 2010, Microsoft Research.

Paryasto, M.W., Kuspriyanto, S. Sutikno and A. Sasongko, 2009. Issues in elliptic curve cryptography implementation. Internetworking Indonesia J., 1: 29-33.

Simon, D.R., 1997. On the power of quantum computation. SIAM J. Comput., 26: 1474-1483.

Subashini, S. and V. Kavitha, 2011. A survey on security issues in service delivery models of cloud computing. J. Network Comput. Applic., 34: 1-11.

Wang, C., Q. Wang, K. Ren and W. Lou, 2009. Ensuring data storage security in cloud computing. Proc. Int. Workshop Quality Serv., 186: 1-9.

Yunis, M.M., 2009. A cloud-free security model for cloud computing. Int. J. Services Standards, 5: 354-375.