Is virtualisation the most secure way to provide shared resources and applications

Marc Reilly

Submitted as part of the requirements for the degree
of MSc in Cloud Computing
at the School of Computing,
National College of Ireland
Dublin, Ireland.

September 2013

Supervisor Michael Bradford

Abstract

This research has been carried out to investigate and propose a method for securing an virtualised environment and to ultimately decide whether virtualisation is the most secure way to provide shared applications and resources. Virtualisation is a very popular mature technology which was first introduced in the sixties by IBM. Lately with the rise of cloud computing people are deploying virtualised environments at a large scale. But a question still surrounds whether these environments based on virtualisation are secure. From researching this topic i have evidence to believe, virtualisation is a secure technology but there are certain areas which still have their vulnerabilities but with the environment and concept proposed it is believed that the security of a virtualised environment can be improved and maintained through intelligent environment design, monitoring and analysis.

Contents

Chapter 1

Introduction

This paper was written to investigate the security of a virtualised enviornment, how secure it is and to propose a method on how it can be secured through proper design,planning and analysis.
Virtualisation is defined by Gartner as:

> *'Virtualisation is the abstraction of IT resources that masks the physical nature and boundaries of those resources from resource users. An IT resource can be a server, a client, storage, networks, applications or OSs. Essentially, any IT building block can potentially be abstracted from resource users.'[6]*

The following is the structure of my review

1. Introduction
2. Literature Review
3. specification
4. Implementation
5. Evaluation
6. Conclusion
7. Bibliography

In order to fully investigate the security procedururures and issues pertained in a virtualised enviornment an extensive review of current issues, best practices and tools

was carried out.

With the growth of cloud computing and the virtual data-centre, virtualisation and security in particular have become a hot topic. Evidence of the growth of the public cloud is shown by a Gartner report predicting the global cloud services market to pass 109 billion dollars in 2012 (19.6% growth) [5]. With this growth comes challenges, virtualised infrastructure both public and private will become more susceptible to attack due to the centralisation of resources. For this reason security is more important than ever.

From reviewing literature on the specified topic, some of the benefits of virtualisation noted have been better resource utilization, efficient disaster recovery, high availability of resources, portability of virtual machines (VMs), isolation of resources and quick and on-demand deployment of resources. But as with all benefits there are negatives.

There are a lot of traditional vulnerabilities which apply to a virtualised environment but there are some risks associated with a virtualised environment that do not exist in a traditional information technology (I.T.) environment such as multi-tenancy and the virtualisation of hardware which can open up new avenues of attack.

Chapter 2

Literature Review

Firstly a review into the current state of the topic will be given using the following sub-categories which reviews literature under the most relevant topics:

1. Guest Security
2. Monitoring
3. Virtual Machine and Snapshot Management
4. Control Access
5. Infrastructure and Software Management

2.1 Guest Security

Guest security in a virtualised environment is paramount. Many sources refer to how an environment is only as secure as its weakest attribute, defence in depth is key. It is important to keep guests logically and physically separated according to Scarfone et.al 2011 [12]. They explain how techniques such as sandboxing can help isolate guest VMs and help prevent attacks such as 'escape' and 'hopping' attacks. Sandboxing creates a logical partition or 'container' which logically separates vm environments and is a useful isolation technique. This is also referred to by Anand et.al, 2012. [1] and Chen et.al [4] emphasises how in multi-tenant environments side-channels and covert channels can open up opening a back door into co-resident virtual machines, a problem which efficient isolation would prevent. According to "Cloud Security with Virtualised Defence and Reputation-Based Trust Management" [8] the logical segregation of guests

can prevent against a catastrophic denial of service attack, again due to isolation. Hsin-Yi, et al.[16] also argue that proper allocation of resources will help choke the Direct Denial of Service (DDoS) attack and only kill one virtual machine. They also argue how migration to different hosts to relieve the absorption of resources can benefit the systems greatly.

Physical partitioning is also a very important factor according to Bellur et.al. [3]. Physical isolation,partitioning and segregation of critical virtual infrastructure is very important in a virtualised environment.*"From the Physical to the Virtual Threat Modelling the Landscape of Virtualisation"* (Lewis, 2012) [10] also refers to this. They stress the importance of not overloading physical infrastructure with critical virtual machines because if a breach of security or a failure was to occur all virtual machines would be unavailable. They explain how carefully planning virtual infrastructure is the most important part of adoption.

However it is clear from researching the area that traditional security measures are also very important in a virtualised environment. All guests must be secured individually and in most cases that is using traditional means such as software updates and anti-virus software.[12] The biggest environment/security procedure change is in the underlying features such as the host hardware and hypervisor. Ideally the guest should not know it is in a virtualised environment. Host security will be discussed further on in the document. Lewis [10] explains how important it is to use threat modelling to identify new risks which virtualisation brings such as image and snapshot management which will be discussed further on.

2.2 Monitoring

Monitoring of a virtual environment is also a key feature. Monitoring is used to locate, detect and predict threats and vulnerabilities in the environment. The ability of the hyper-visor to monitor the environment through its introspection capabilities is key. Scarfone et.al 2011 [12] state how important it is for the hypervisor to be aware of its surroundings in order for it to effectively monitor the environment. The host must be fully aware of its guests at all times in order to effectively secure the system. Ueno et.al 2010 [18] also stress the importance of hardware transparency for the efficiency of the system.Logging and heuristics can be a great tool for making hosts aware of its environment. Kelley [9] mentions how Logging is an invaluable resource.All events should be logged and monitored in order to detect and diagnose failures. This feature is a necessity in large automated virtualised environments as physical monitoring and intrusion detection would be impossible.

One area highlighted as a problem in this area is the effectiveness of the monitoring of virtual networks. Here physical networking rules are not as relevant as they do not apply within the virtual environment. Intra-VM traffic is seen as a problem and cannot be monitored by the physical network. This type of communication is hard to monitor and Kelley [9] explains how a "blind spot" may exist in virtualised environments. She states that intra-vm traffic can be carried out without using the physical network, therefore it cannot be detected by traditional network monitoring tools. This can become a big problem within domains, more specifically Trusted virtual domains(TVD) [7]. According to Tupakula et.al. (2011) [17], Here VMs can freely communicate with each other within domains once the host/TVD master is compromised. They say how security concepts or tools implemented such as "honey-pots", intrusion detection and prevention systems can be compromised once the "master" is compromised. This infection would be catastrophic in a fully virtualised environment. They proposed policies to deal with these kinds of attacks- "TVDSEC" [17].

There are still physical network threats which can affect a virtual environment. According to Chen, et al., 2010 [4] web security is still a big issue in virtualised applications. As most virtualised applications are accessed from web browsers, they can be susceptible to traditional internet based threats such as phishing and cross site scripting. In the past very sophisticated attacks have occurred most notably Amazon. [15]Here attackers recorded the IP addresses of virtual machines they deployed to try and figure out if certain IP addresses matched to where they were located physically in the virtual infrastructure. From here, if they wanted they could get deployed on the same physical server as whatever other customer they wanted, as long as they knew the other users IP addresses. This method would give a possible attacker a base or a point of attack against another user.[9] To counteract this techniques such as dynamic reputation systems have been developed. These systems monitor user activity to build a reputation score to which customers are rewarded and if necessary punished.

Monitoring is also very important in ensuring the security of the environment. Tools such as load-balancing depend highly on monitoring of resource usage. The Application of load balancing techniques can greatly reduce the risk of service unavailability and help prevent against denial of service attacks and Economic denial of service attacks.

2.3 VM Image/Snapshot Management

Due to the on demand and data centralization characteristics of virtualisation it is very important that vm images and snapshots are managed securely and effectively. VHDs or virtual hard disks are the logical partitions used by virtual machines. They are

stored as files and contain all information and files related to its virtual machine. In the literature a few best practices and threats to the security of these were proposed.

One of the main points raised was "server sprawl". This is the practice of provisioning an excess amount of servers that is actually needed. This is a very big problem in virtualised environments due to the ability to provision resources on demand rapidly. In "How Data-Centric Protection increases security in cloud and virtualisation" Kelley [9] notes the absence of overheads in providing extra resources in a virtualised environment. The Administrator can simply provision new resources without having to pay and wait for the delivery of extra physical resources. This is a big contributing factor to server sprawl- the elements of rapid, on-demand provisioning of resources. Lewis [10] and Scarfone [12] stress that sprawl can cause over-utilisation of resources and un-manageability of the environment which may lead to catastrophic failure and degradation of systems. According to Anand et.al (2012) [2] sprawl increases the management complexity of the environment and over time issues will arise.[12] Some of the issues which may affect the security of the system are transparency and traceability e.g. when System-wide updates are carried out some VMs may be missed and will be incompatible with archived clean images leading to failure/unavailability upon recovery.Sabahi et.al. 2011 an [11] proposal a least-access policy to control the provisioning of virtual resources. This policy would trust the responsibility of provisioning VMs to certain administrators only leading to controlled and moderated deployments. Lewis [10] also refers to strict vm deployment policy to control sprawl.

Another aspect of efficient image and snapshot management is ease of recovery. Virtual machines can be reliable when fast system recovery is needed. Scarfone et.al [12] discuss the procedure for handling with compromised VMs. They proposed that organisations should have a 'gold image' repository. This repository would store clean clones of virtual machines for recovery purposes.Upon failure a compromised vm would be deleted and a user session would be restored from a clean image. This practice would be very useful for quick redeployment of resources upon failure. According to Semanian et.al. 2011 [13] Virtual machines can be easily replicated and stored as back-ups. They can be migrated across physical servers, suspended and resumed upon request. This can provide fault resilience to a system at levels traditional resources could not. A working copy of a virtual machine/application can be deployed within minutes. If a traditional node has to be restored it can take quite some time. However it must be noted that according to Lewis [10] migration of infected VMs to other hosts can be very dangerous and can easily spread viruses and malware. Therefore if a 'dirty' image is detected it must be shut down and isolated immediately in order to prevent spread of the virus and a 'gold image' spun up as a replacement.

According to Kelley [9] these clean image repositories can also cause a security risk as virtual machines are stored as files on the storage system they can be accessed at any time if the host is compromised and easily exported to a rogue source. Therefore techniques are needed to protect these VHDs which are at rest. Lewis [10] proposes physically separating these repositories while restricting access. He also proposes encrypting these repositories so if they are infiltrated they cannot be used without the keys. Another issue with these repositories could be bad update policies. According to Lewis [10] this technique is only secure if proper patches and management is in place to take care of these gold images. If another version from the repository is deployed to replace a infected production vm, incompatibilities may be present and that will present longer downtimes and unavailability. He therefore encourages administrators to carry out scheduled updates. He also advises administrators do this at off-peak times and spread out the updates to prevent "update-storm" which would temporarily slow down the system during the maintenance times.

Virtual desktop infrastructure is another very popular virtualisation technology. This gives administrators the ability to provide centrally managed desktop environments. Here all data is stored on the host environment and not the physical client. Lewis [10] explains how this can be a huge security benefit over traditional deployments as no confidential information can be stored locally ensuring data security. He also stresses the benefits of virus and malware protection.He explains how non-persistent images can be deployed to users every time they access the system. This means a fresh 'gold copy' can be distributed to every user every time they log on. He explains how this can prevent the spread of vicious malware and viruses within the domain. Shackleford [14] also recommends the use of VDI over traditional models for security reasons. He also discussed how the emergence of bring your own device (B.Y.O.D.) in organisations this will become more widespread due to its ability to prevent local storage on employee devices.

2.4 Control Access

The security of the hyper-visor is the a very high priority in a virtualised environment as if it is infiltrated it affects all its guests and the attacker will have full control. Therefore we must have control access.

Most articles refer to policy and best practice to secure the hyper-visor. Scarfone [12] proposes that only a small select group of administrators have access to the Virtual Machine Manager to reduce risks. They also proposed that hardware token based authorisation be implemented. They propose remote administration only be carried out

on trusted networks with strict firewalls.Anand et.al [2] propose the implementation of a dedicated administrator/management network for security reasons using a virtual private network (VPN). They also propose strict access policies to the VMM with administration only access and read-only access to users who need it.

Most articles stress how important it is to secure each component of the system of the system individually to ensure "defence in depth". Ueno et.al [18] explain the importance of defence in depth and how a system is only as strong as its weakest components. They explain how one must reduce the number of attack vectors and points of weakness in the system and the best place to start is at the hyper-visor level and who can control it. Shackleford [14] cites that even a default configuration file can be a weakness and that all settings should be changed be changes to suit the deployment environment.

Physical security is also a contributing factor with virtual environments as with traditional. Scarfone [12] explains how this factor is often overlooked. All an attacker would need is to be in the same room as the physical Host server to do damage to the system. Therefore physical access control should also be implemented in a virtual environments as it would in a traditional environment. Also the use of automation in the management of virtualised resources greatly reduces the need for human access to server rooms and therefore removes another potential error. In today's large data-centres there are around one full time staff members per one hundred physical homogeneous servers.

2.5 Infrastructure and Software Management

Efficient management of the host infrastructure and software is vital in securing the virtual environment. One of the main points which came up in the literature was the allocation of shared resources. Many of the sources explained how it is important to reduce the number of attack vectors in a virtualised system.Scarfone et.al [12] cited that all unused hardware on the the hosts and clients must be disabled and/or removed to reduce the number of areas an attacker can exploit. The example they used was removable disk drives and network interface cards (NICs). These could be vital attack areas for an attacker to gain access to the system. They also proposed that all unused virtualised hardware and hyper-visor features which are not being used be removed. If left on the system and unused their security may be neglected and therefore they would be a weak-point i.e. a attack vector.

Another key point made was the implementation and design of hyper-visors to prevent attackers detecting the hyper-visor/host os type/make or version. This kind of

information can be used maliciously. This would give attackers access to other VMs data and activity. To do this attackers identify patterns in users activities and try to find holes into the system from this or even clues to what kind of activities they are performing. Semnanian et al.[13] refer to two techniques used to attack virtualised resources called hyper-jacking and a root kit. Both these techniques attack the host by accessing the underlying physical hardware through gaps discovered in the network. They also stress how it is necessary to adequately secure VMs before deployment. This can reduce the risk of worm, Trojans and viruses spreading. Shackleford [14] explains that hyper-visor detection can reveal information such as what other machines are on the system. He mentioned how VMmalware is becoming more prevalent with techniques such as "stormworm" in Microsoft virtual PC environments which will look for trails of virtual machines in the system. There are other variants such as Phatbot which have vm detection. Scarfone [12] also references this topic suggesting that better hyper-visor design is needed. They also suggest changing or customising the hyper-visor to remove common identifiers which may help attackers. Anand et.al. [1]suggests that hyper-visors must be more aware of compromise in order to deal with these threats. Further evolution of intrusion detection and prevention systems would help greatly here. They believe isolation of the host and guests must be ensured and consistent in order to secure the environment.

In the article "Virtualization Technology and its Impact on Computer Hardware Architecture" [13] they propose an intelligent monitoring tool to detect intrusions and suspect behaviour from users using logs to map their activities. This is also suggested in Trusted Cloud Computing with Secure Resources and Data Colouring [8] they propose a trust and reputation management system which protects VMs from software based attacks by using watermarking, encryption and certificates to grant access to a trusted zone in which the user can interact with the VMs.

Another key point made was in regards to the removal of unused software from the hosts and clients. These can also be useful attack avenues. Scarfone [12] suggests stripping the host Operating system to the bare minimum. This would involve only keeping core tools and utilities which are necessary for running the environment on the host. Anad et.al [2] also refers to this topic. He says that if a feature is not necessary you must remove it.

Ensuring the integrity of the system is also very important as explained by Scarfone et.al [12]. They explain the importance of having the host infrastructure sync with trusted time servers in order to keep storage and other tools such as meta-data accurate. Sabahi et.al [11] states how important it is to deploy integrity checking

tools in the virtualisation environment due to the usual mass centralisation of shared resources and not doing so can provide a very large data security risk.

Shared resources are one of the key motivators to virtualisation, however according to the literature it can be a big threat if its not managed and controlled properly. According to Sabahi [11] this is becoming an even bigger problem in virtualisation due to the rise of cloud computing and more specifically public 'shared' clouds. Here large datacenters are populated with commodity hardware and use is sold as a utility. These hardware resources are shared between users on a multi-tenant basis. Sabahi highlights a few issues which may arise such as data remenance. To counter this he proposes encryption and strict key controls on data stored on shared storage. If implemented, once data and the key is deleted, even if fragments of data are still present they cannot be recovered. Another problem they identify is the dangers of data leakage in shared storage and how dangerous that can be. Again it may not be preventable but the data can be protected with encryption. A third issue they raised was the recovery of client encrypted data on the public cloud. If encryption is enforced on the client side and a host failure occurs in the cloud data will not be recoverable. This would be catastrophic for a client.

Many documents such as Anad et.al. [2] and Scarfone et.al. [12] suggest that access to shared resources such as shared clipboards and storage should only be on a least-access policy i.e. guests only have access to what they need. This policy would reduce the number of attack vectors. It is also noted that resources should be physically and logically separated into how critical or how regulated the information is e.g. HIPAA regulated data should be physically and logically segregated from open-source information storage to reduce risk.

It is important to point out that with virtualisation a large amount of virtual infrastructure and software is deployed and can be difficult to manage and update. As cited earlier Lewis et.al. [10] proposes that all updates, back ups and patches should be centralised and scheduled. This in theory would make the virtual environment more manageable than the traditional one due to centralised automated management. However they do cite some issues to be aware of such as boot storm, scan storm and update storm. These refer to mass activity in their relative areas which cause inefficient use of resources and lagging responses. One example of boot storm would be booting all virtual machines up at nine o clock on a Monday morning, this would cause delays and a strain of resources. He proposes scheduling these activities at spread out off peak times. It is also noted that non-persistent VDI deployments can help with widespread

patches to systems and software.

It is important to plan ones virtual resources and applications carefully as vendor lock in may occur. In the article by Hwang et.al [8] they talk about using cross platform standards while implementing virtualised environments. The reason for this is vendor lock in can lock in resources and applications to certain providers with no way of retrieving them. They propose a standard set of APIs called the Open Virtual Format (OVF) which would enable efficient security software distribution an encourage virtual machine mobility. This would standardize the competition and eliminate vendor lock in.

Chapter 3

Specification

From reviewing literature on the topic it is obvious that there is an extensive amount of material available on the subject and it is a very popular topic. It is obvious that a lot of the problems faced in this area have adequate security precautions already available but some security features are still in their early stages. However with any technology which is growing in popularity at such a rapid pace, vulnerabilities will be exposed. But it is evident that in order to secure a virtualised environment extensive event and usage monitoring and analysis must be put in place in order to enforce security policies for the environment.

In light of the research carried out, the goal of this research is to gather data on the virtualised environment in a way in which it can be easily retrieved, processed and analysed in order to provide more in depth insights into the environment. As shown from reviewing the relevant literature the improvement of processes and policies is vital in ensuring a secure environment. In making the environment more auditable and system timelines more easily traceable too, it is believed better decisions can be made into controlling and hence securing the environment. The goal is to provide a concept whereby a system administrator can more effectively access, assess and manage the security needs and status of a virtualised environment.

To help organise and collect data in such a way requires a lot of planning and analysis of the environmental needs and skillset of the administrator. To help wage this, a number of implementations were analysed and tested on three main parameters

Complexity

Cloud deployments can scale to very large sizes and reducing complexity can be a very difficult task. In order for system administrators to effectively secure these environments the complexity should be reduced to help remove potential security vulnerabilities. It would be very difficult to remove all complexity from a cloud environment, therefore it is important that any complexities are kept for business critical processes or functions and not focused on support activities such as log retention, collection, management and analysis. Although, as evident from the previously described research, security is also a vital aspect of any cloud environment which must be maintained with minimum difficulty and overheads. However, if a solution can be chosen which can greatly reduce the difficulty in managing and scaling the process for gathering system information and activity it should be seriously considered.

scalability

To achieve reliability and high availability in a cloud environment any solution must be able to scale rapidly and effectively with the cloud environment in order to be affective. The elastic and on-demand characteristics of a of cloud environment can have a huge impact on the systems used within it. Another key factor is how scaling impacts the implementation in terms of its effect on the underlying system. If scaling the cloud environment has a negative impact on the performance of a monitoring system this can greatly degrade the performance of the environment and prevent the monitoriong system from accurately reporting vital information.

management

In order for a monitoring implementation to have a positive effect on the environment management overheads must be minimal. If the system causes distraction from the core business function the environment would not be efficient, economical or feasible to implement. The system administrator should not have to spend valuable time managing a monitoring subsystem . The valuable time saved from a system which requires minimum management time could be spent trying to harden the system further using the collected data. Another key benefit of a low management overhead is that system administrators could focus on innovation and system upkeep, this is a key factor in why the final solution was chosen.

The following section will list and explain the tools analysed and tested for the

environment. The following systems were implemented and tested to help achieve the best possible mapping to the three main points in the selection criteria above.

1. Rsyslog
2. Ganglia
3. Logstash
4. Elasticsearch
5. Kibana
6. Rapidminer
7. Eucalyptus

Rsyslog

Rsyslog is a log collection and forwarding utility for Unix environments based on the syslog protocol. It is a widely used method of gathering and forwarding Unix system logs. It provides features such as filtering and log forwarding over TCP and UDP. A typical use case using rsyslog a system gathers log files on a central log server using the supported TCP and UDP forwarding functions. This can provide many benefits such as a single point of contact for retrieving and viewing system events when needed.

Ganglia

Ganglia is used to gather data on both windows and Unix systems in distributed computing environments. The ganglia monitoring system consists of three components:

- Ganglia Monitoring daemon
- Ganglia Meta Daemon
- Ganglia Web Frontend

Ganglia Monitoring Daemon This is the monitoring daemon (gmond) used to collect metrics from systems it is running on. In order to collect metrics in a cluster the gmond daemon must be running on all nodes in which data collection is needed. The gmond collect user specified metrics and forwards them using either TCP or UDP

to the Gmetad. Multiple destinations can be configured in the gmond.conf if needed. The transport of data may be done through multicast or unicast broadcast. The data is broadcast in xml form on a user specified port. The data is then aggregated and parsed by the Gmetad.

Gmetad The gmetead collects, aggregates and stores metrics from a cluster of nodes running the ganglia monitoring daemon(gmond). After data is collected it is parsed and stored in a round robin database called RRDtool. The RRDtool is designed to store time series data but also provides built in tools for graphing data and calculating metrics using mathematical functions. This function is utilized by the ganglia web frontend. Each metric is stored in its own individual table and metrics are graphed and calculated on demand. This feature is extremely useful to the ganglia web frontend as it reduces the burden on it to graph large amounts of data on the frontend.

Ganglia web frontend The ganglia web frontend is used to gather and display metrics from the cluster. Using PHP the ganglia web frontend gathers the data using the RRDTools built in functions which calculate metrics and build graphs. These graphs are then rendered onto webpages which can be organised by metric, node or cluster. This feature is a very useful one however it does it will only provide an administrator with a high level view of the enviornments current status, that is to properly secure a system these metrics would not explain the entire situation on their own. To gain a better understanding, environmental events need to be matched with these metrics to gain a meaningful insight. Also as the cluster scales metrics and graphs for the environment can become quite cluttered on the ganglia web frontend. This can cause unnecessary complexity for the system administrator and make it quite hard to gauge how the system is performing and how the system can be secured.

Once the the log files and system data has been collected one must find a way of analysing them. In a large distributed virtualised enviornment this task can be very difficult due to the size and in some cases the geographic distribution of systems in the enviornment. Therefore an effective way to ship and aggregate logs for an entire virtualised enviornment into a central location can be a difficault process. To help combat this issue the following tools were investigated:

Logstash

Logstash is an open source tool developed by James Turnbull. It is used to collect, parse and store log files. It has three main features

- Input
- Filter
- Output

An input is where your data is taken in by Logstash. There are over thirty five official inputs developed by the Logstash team including ganglia, syslog, file, sqlite, pipe, log4g etc. There are also many inputs developed by the Logstash community and submitted for public use. When defining an input you specify certain parameters in your Logstash configuration file. Each input can contain different parameters but most share a few common parameters such as tag, source host, port and type. These parameters are very important in routing, tagging and retrieving data from hosts and are very useful in large virtualised environments as all data can be received centrally on a specified port, parsed and tagged before it is stored. This can help in retrieval of information as all data can be sorted into categories using tags and it helps reduce the complexity of parsing the unstructured data into more organised meaningful data for use by the administrator.

Filters are used to help structure the data after parsing. Examples of filters are grok, date, alter, mutate and grep. Again these parameters are specified in the configuration file. The purpose of this feature is to allow users manipulate data with ease after it has been tagged and parsed. This allows you to structure data, remove variables which are not needed or even just change the format of the data. An example of this is the grok and date feature. Grok allows the user to remove unneeded data or even restructure the data into a desired structure. Another example is the date feature can be used to parse timestamps into different formats.

After the data is passed though the filter it moves onto the output phase. Here Logstash passes the data onto whatever endpoint specified by the user in the configuration file. The output feature is how Logstash interacts with the storage subsystem. Again there are a large number of outputs developed by Logstash such as AWS S3, AWS simple notification service, Ganglia, Nagios, elasticsearch, pipe, Redis and a file output.

Elasticsearch

When choosing a storage log solution for a cloud environment it is important that the solution is highly scalable, easily maintainable and simple to use as outlined earlier. These three features were key in choosing the underlying storage system. It was recognised very early on that one of the most important considerations to keep in mind was how would the system scale over time and how easily could it be queried.

Elasticsearch is a highly scalable open source Java based search server based on the Lucene project by apache which allows users carry out real time search analytics. It operates in a master slave configuration and allows users to insert data in JSON format and then query it using a RESTful interface. The main features of elastic search are An index is where the data is indexed and stored, similar to a table in relational database terms. There can be many indexes in an elastic search cluster and data is optimized for search and retrieval. In elasticsearch, a document refers to a line of data inserted into an index. When storing documents elastic search takes advantage of a feature called sharding which will distribute document among different nodes in the cluster making it data highly durable. It is very important to note that elasticsearch does all this automatically therefore reducing management overheads and complexity.

Another key feature of elasticsearch is its simple scalability. Again this can reduce complexity and management providing system administrators with more time to focus on managing the systems in an organisation. To scale out an elasticsearch cluster you simply add a node to your cluster and run elasticsearch specifying the name of your cluster and whether the node is a master or slave. Through multicast using xen disco discovery the clusters master node automatically recognises the slave and starts sharding data to it. Sharding helps improve the speed and efficiency of writing and retrieving data as all data is written and retrieved in parallel. Each index is responsible for sharding documents to the slave nodes from the master. The use of sharding can also greatly reduce the load on the cluster and it will be distributed among all the nodes in the cluster.

Elasticsearch is queried using a RESTful interface which provides a lot of flexibility in how data can be queried and retrieved from the cluster. Custom interfaces and applications can be developed to interact with Elasticsearch if necessary providing a wide variety of use cases for deployment and utilization which can be taken advantage of if required.

Kibana

Kibana is a web based interface which interacts with the elasticsearch RESTful interface to retrieve data from elasticsearch. Using this data it uses data visualisation and analysis techniques to help administrators gain a deeper insight into the activity and performance of this systems. Users can query elasticsearch using the search filed provided using normal text queries or through the Apache Lucene syntax.

When search results are returned users can display, omit and sort data according to the tags specified by the Logstash configuration. This can help in focusing on specific metrics and data points. Results can also be graphed and mapped out using Kibana. This can help provide a better insight into the data and help improve transparency. This is very important as it reduces the complexity as it provides a single point of contact for all system logs and system usage in one interactive interface. Kibana also provides basic analysis of the data with features such as trends, term frequency and statistics such as max, standard deviation, mean, min, count, sum of squares and variance on demand which would be a very useful tool for analysing system usage data from ganglia.

Rapidminer

Rapidminer is an open source application which provides data mining and machine learning tools. These features are provided through a drag and drop graphic user interface. It provides some standard features for analysing and manipulating data but also provides in house and community contributed extensions providing extra features and analysis techniques. It makes use of features from the Weka machine learning environment from Waikato University in New Zealand and the R-Projects schemes for modelling statistics. This application can help provide an environment for further analysing the data collected from the system log files and useage. Through analysing this data the system administrator can harden the systems security by gaining a more meaningful insight into how the system performs. One such feature is the Support Vector machine (SVM) algorithim. SVM is a linear classifier which means is classifies the data based on a linear combination of the values provided in the dataset. The classification of the collected data can help identify certain characteristics and trends in the dataset. In this case it will try to accurately classify the data in order to provide valuable information on how secure the system is and what characteristics in the collected data usually lead to system weakness and insecurity. There are also

other models for classification of data such as decision trees and the k-nearest neighbor algorithm (K-nn) Classifier which can also be implemented.

Eucalyptus

Eucalyptus is a open source cloud computing environment designed to help provide highly scalable compute, storage and networking. It has four main features called

Cloud Controller The cloud controller is used to manage the cloud enviornment by hosting administrative tools and dealing with the clusters API calls, sheueling and system mmonitoring. From here a system administrator can manage the whole cloud environment through tools such as the web interface, eucatools and the admin console.

cluster controller The cluster controller helps manage the storage and compute of the cloud enviornment by communication with the storage and node controller.

Storage controller The storage controller helps manage eucalyptus block volumes which are inpersistant storage volums which virtual machines are run on. It also communicates with the cluster and node controller.

Walrus The walrus provides persistent storage for the cluster and stores virtual machine images and snapshots persistently. It can also be used as an object store for the cluster taking advantage of HTTP put and get requests.

node controller The node controller is used to run virtual machines and manage the networks. It is here that the hypervisor is installed

Chapter 4

Implementation

A test environment was set up with a total of six virtual machines run on a VMware esxi 5.1 host and one amazon web services elastic compute cloud. These seven virtual machines consisted of

- One standalone eucalyptus host was set up with a KVM hypervisor
- One network address translation (NAT) instance
- One Central Log Collection server
- Three elasticsearch nodes which consisted of one master node and two slave nodes.
- One EC2 m3-2xlarge windows server 2008 R2 instance configured with Rapidminer

The eucalyptus instance was set up on a centos operating system. The KVM hypervisor was then installed by taking advantage of nested virtualization. Eucalyptus was then installed and configured using the standard configuration. Since the researcher had no control over the virtual machine network a NAT instance was also set up to provide an extra layer of security but also to have more control of the underlying network and provide more flexibility.
Two Debian virtual machines were then created and instantiated in the Eucalyptus envioronment. They were given IP addresses on the NAT network to enable them to communicate with instances outside of the eucalyptus environment. The eucalyptus host and its guests were then configured to forward log files using rSyslog over UDP. For security reasons the logs from the Debian guests were forwarded to the eucalyptus host for aggregation before being forwarded onto the central log collection server. It is best practice when utilizing central log collection that the logs are hosted in more

than one location in case the system gets compromised and they are tampered with. This is often used by intruders as a way of deleting evidence of intrusion. Also if the eucalyptus host lost connectivity with the rest of the network and the Debian guests werent sending logs to the eucalyptus host no logs would be saved as the central log server would be unreachable and traceability of the guests would be compromised as all logs forwarded would be lost. The ganglia monitoring daemon (GMOND) was then configured on the Eucalyptus guest virtual machines and the eucalyptus host to forward metrics to the central log server.

The central log server was configured with Ubuntu server 12.10. On this instance rsyslog was configured to receive all log files on UDP port 514 and then forward them to Logstash which was running on the elasticsearch cluster. This node was configured to act as an intermediate between the cloud environment and elasticsearch. This was to ensure that if in the unlikely event that the elasticsearch cluster would become unavailable that the log files were still being retained and could be forwarded at a later stage. Ganglia was also configured on this instance. However unlike the eucalyptus node the ganglia meta daemon (gmeted) along with the gmond were installed and configured here to collect metrics from the cluster so they could be forwarded to elasticsearch. To do this, ganglia was configured to send metrics directly to central log host on port 8649.

The elasticsearch master node was tested on two environments which were on private VMware esxi hardware and also on ec2. Since elasticsearchs automated configuration of slave nodes uses multicast, some configuration changes were needed for elastic search to run on ec2 as AWS does not permit multicast in their cloud enviornment due to security reasons. Elasticsearch was run on ec2 but it was concluded that for ease of scalability the public cloud environment was not suitable due to complex configuration. While on local hardware elasticsearch scalability is not hindered as multicast communication can be used.

Logstash was also configured on the elasticsearch master node to gather the logs from the central log server. In the configuration file the inputs, filters and outputs were specified.

For the inputs the Logstash plugins for ganglia and UDP were used. For the ganglia plugin the port to receive metrics was specified as port 8649 and the type was set to ganglia. The UDP port was set up to retrieve inputs from the central log server utilizing rsyslogs message forwarding feature on port 5000. Its type was set to syslog. The type parameter helps categorise the inputs when storing them in elasticsearch for a more simplistic and accurate retrieval. Specifying the type also helps identify inputs when looking to pass them to certain outputs. For example if there were many outputs specified to different endpoints, the user can specify what data gets output to a specific

endpoint by selecting a specific type.
After data is taken in using the input plugins it can then be passed to either a filter or an output. In this implementation the syslog messages are passed to a filter to help parse them more accurately and format timestamps to a correct and uniform format. Ganglia messages are passed straight to the output as they are already correctly formatted and parsed by the input parameter. In order for the filter to collect just syslog messages the type syslog must be specified. The filter implemented ensures a standardised filtering of the data before it is sent to the elasticsearch cluster. It must be noted that is is regarded best practice to carry out as much pre-processing of data as possible as it is being streamed into the system as it reduces the workload in analysis jobs later on in the dataflow/workflow.
After data is filtered it is passed to the output phase of Logstash. Here data is streamed into the "elasticsearch http" plugin. This plugin receives the json data from the filter and plugin and sends it to a specified elasticsearch host on a user specified port which, by default is port 9200.
This plugin communicates with the elasticsearch master node by streaming data to the host using its RESTful interface. Another parameter which is vital in the "elasticsearch http" output is the flush size. Here you must specify how many documents are flushed to the elasticsearch cluster. From testing this feature on the current setup a flush parameter of 50 documents was seen as the most reliable. If no flush size was specified, the elasticsearch cluster would fail to index the documents being streamed to it by Logstash and the system would stop streaming the log files to elasticsearch. This flush problem is mainly due to the vast amount of system resources which can be consumed if the flush processes are uncontrolled and sporadic
When elasticsearch receives the documents through port 9200 it inserts documents into a dynamic index which is sorted by date in the format Logstash-yyyy-mm-dd. This can help in organising system log information or documents as they are referred to in elasticsearch. Dynamic naming of indexes can also aid easy archiving of data if necessary. In regards to storing and sorting of documents Elasticsearch does all the hard work. Elasticsearch will sort the documents by type and unique key. Elasticsearch is schema free so the user does not have to specify the data type or key of documents as elasticsearch will do this automatically.
The simplicity and elasticity of running an elasticsearch cluster is a big selling point for the storage and search system. Other key points to make are how it can enable real time analytics of documents. Once a document is sent to the cluster it is searchable. This can help an administrator identify system vulnerabilities instantly with the right tools available to them. The elasticity concept is also very important in a cloud environment as the cluster can be scaled up and down with ease. It is designed to scale

horizontally, if more capacity is needed the administrator just needs to add a node, start it and elastic search will reorganise itself. Internal Load balancing techniques can also be implemented to ensure that as the cluster scales it can still cope with massive amounts of data. This can be done by having dedicated master nodes which distribute load among the slave nodes, again this is an inbuilt feature.

Other features such as high availability and redundancy can also be taken advantage of. Replicas can be configured within the configuration file. Also if a elasticsearch detects a failed node it will remove it from the cluster and direct documents elsewhere. Elasticsearch also has a version control feature which can help in large environments where massive amounts of writes are happening at one time. This feature does not really apply to log files as the systems producing the logs would only write one event.

All the features mentioned can greatly increase the security of a system if implemented effectively. When logs are gathered and stored it is very important that they are kept in a secure environment and in a cloud environment it is important that the environment that the logs are being stored in is resilient and scalable.

As evident from the research earlier, the log files are not very useful unless the system administrator uses them effectively. Key insights into system usage can be gathered from these documents to help improve system security and reliability. After the gathering and storage phase comes the visualisation and analysis phase. The goal of this phase is to provide a system administrator with a single point of contact to analyse the system in real time and in the longer term analyse how the environment is being used, how it can be improved and ultimately how can it be secured.

To aid with visualisation and analysis two tools were used which are kibana and Rapidminer

Kibana was set up on the central log server to help search and analyse documents on the Elasticsearch cluster. As described earlier Kibana is a tool developed in ruby which allows you to visualise and search elasticsearch documents. Its use in this setup is to provide real time on demand insights and analysis of the elasticsearch cluster.

This tool can make looking for issues in an environment a lot more straight forward and insightful. Using the search bar in Kibana you can query the elasticsearch cluster using keywords, dates, types or statements using the Lucene syntax. By centralizing the log files in elasticsearch and providing a interactive web based graphical user interface the system administrator can query log files for the entire virtualised environment from one place. For brief analysis of the environment or debugging this is ideal as it provides instant access to the log files.

In terms of analysis Kibana will display metrics on different variables or categories that were configured in the elasticsearch indexes, for example it will show the most common system events which cause errors by selecting the error status in the log files over a

certain period and then selecting trends the administrator can see what error status codes are appearing the most and what system events are causing them. This feature is also very useful in a web hosting environment where an administrator can query apache logs to see where users are accessing the website from and identify access trends to help improve the websites performance. Being able to view this in real time would be a vital tool in certain scenarios such as a direct denial of service attack (DDoS) where a quick reaction would be needed to help prevent downtime and ultimately a disruption to business activities.

To perform a more in-depth analysis on logs files (documents) Rapidminer was used. The use of this tool differs from the analysis use case of Kibana as it is seen that Rapidminer be used on a periodic basis instead of being used for real time insights. This is due to the fact that large datasets can take quite some time to analyse and gain sights from them.

The analysis process in Rapidminer starts with importing a dataset into the repository. To do this you can use one of the "operators" provided by Rapidminer. Operators are predefined tools and algorithms used to build the overall analysis process. Using the graphical user interface in Rapidminer you can simply drag and drop operators into the main process workflow. The workflow helps give a graphical representation of the entire process and is a very helpful tool for less experienced or novice users. This again helps reduce the complexity of scripting and developing as you would in traditional data mining.By using the "import csv" operator the user can import a CSV file exported from Kibana. Here a large collection of documents were retrieved from elasticsearch by using the export function in Kibana which allows users retrieve documents in CSV format. Once the export from Kibana is done the data is then imported into Rapidminer for pre-processing and analysis using the previously mentioned import csv operator.

After records have been imported into the Rapidminer repository they must be pre-processed in order to improve the accuracy of the analysis and remove any outliers, inaccurate or unneeded data. For example, when using support vector machine algorithm the following must be heeded

- One must not use polynomial or nominal attributes, attributes must be numerical
- One may use a numerical or binominal label but not polynomial or a one class label
- Also missing values or unlabelled data is not permitted.

From first glance it is easy to see that the log files from elasticsearch are not suitable as all attributes are not numerical and the data is mostly unlabelled. Therefore pre-processing of data is necessary in order to achieve the most accurate results.

Firstly the a role for each of the variables must be set in order for the support vector machine to identify the different variables. For support vector machines there must be one label attribute which classifies the data and many attributes which help define the label. This is done by using the select role operator in Rapidminer.
After the user has specified the label and attributes the dataset requires more pre-processing. As described earlier support vector machines must use numerical attributes. This is a clear problem as log files are written in human readable text format. Therefore a Rapidminer operator must be used in order to convert the text attributes i.e. log messages into numeric attributes. The operator used is called "nominal to numeric" which will map all non-numeric attributes in user specified fields to a numeric or binary attribute in order for the support vector machine to carry out its analysis -In other words each text based message in the dataset will be converted to a unique identifier or column name and if the message or identifier appears in a log message the value will be set to one and If not, the value will be set to zero.

Once these steps have been carried out the pre-processing phase has been completed. The data set must now be passed to the support vector machine to allow it to analyse the data and build a classification model for classifying the attributes in the dataset.

To do this the support vector machine takes the input and it analyses each log message and tries to learn how it can be classified. As stated earlier the support vector machine allows the user to specify one or two classes as the label characteristic. Therefore in this example the support vector machine will analyse each log message individually and try to determine how it is classified into one class or the other based on the user defined labels. After the analysis is complete the support vector machine will produce a model in which it will specify the weight of each attribute. This weight is an indication of how influential the attribute is in the classification process and where it is situated in relation to the hyperplane. They are represented as a positive and negative integer.This weight is an indication of how influential the attribute is in the classification process and where it is situated in relation to the hyperplane.

The hyperplane is a boundary which helps distinguish the attributes between classes .The success of the training model is indicated by the generalization error. The generalization error is the measure of how far the learner models predictions are from the teaching model.
Afer the support vector machine has created a training model its performance must

be measured in order to see how successfully the learning model can apply attribute classification. To do this the user must use the model created from the support vector machine and test it against a unlabelled dataset. In Rapidminer this can be done by using the apply model operator.
The apply model operator takes the output of the support vector machine operator (the training model), tries to learn it and then attempts to classify a user specified unlabelled dataset to test how the model performs. This process gives an indication to how accurately the support vectors machines learning model can be applied.

To measure the accuracy of the model created the validation operator is used. The "validation" operator takes the output of the apply model operator and measures how accurately the support vector machine learning model was applied to classify the unlabelled dataset provided. The result from the validation is a metric calculating the mean squared error. The accuracy of the classifier is calculated using the mean squared error of the line. This calculation shows how accurately the learning model classifies the unlabelled attributes against the labelled attributes

Once these values are retrieved the researcher can gauge how accurately the support vector machine can classify log messages and system usage into classes which can be used by the administrator to improve system security and in general track system usage and vulnerabilities.

Chapter 5

Evaluation

In order to evaluate the proposed implementation a number of tests were carried out on each element of the environment to ensure they could reliably perform their assigned workloads. It is very important that each element performs accurately and reliably in order for the results of the analysis tools to be accurate and trust worthy.

In order to evaluate the performance of ganglia its scalability and reliability were investigated. It is vital that every element of the environment is scalable due to the nature of a virtualised environment. With ganglia, monitoring daemons were deployed on ten virtual machines operating on multicast sending data back to one gmond. The virtual machines were then stress tested in order to see how they coped under pressure. Systems were also stress tested to make sure metrics and spikes in resources were reported accurately and Ganglia did not show any signs of unreliability under stress. Ganglia did not show any signs of weakness or unreliability. This outcome was expected as ganglia is a mature application being used in many large distributed environments around the world.

The same results were recorded while testing rsyslog and again as expected it performed without any issues. Rsyslog is another widely adopted tool which ships as standard with most Linux distributions therefore it has a proven track record of being deployed in large virtualised environments.

Logstash's performance was evaluated by supplying vast amounts of data to the input plugins to test how it accurately it could operate. The inputs operated as normal, however for outputs to elasticsearch the configuration needed some tuning as the system resources were unable to cope with the document flush size as stated

earlier. If Logstash attempted to flush too many documents to elasticsearch in one process, elasticsearch would fail to index the events and the documents would not be stored in the elasticsearch cluster. It was observed that a flush size of fifty was the optimal parameter for the current setup. Once the flush size was set to fifty Logstash did not encounter any issue in receiving inputs or providing out puts to elasticsearch. It must also be noted that deployments with the Redis database were also tested but were judged to be surplus to requirements in the current setup. In that setup the Redis database was being used as a queue for Logstash to help relieve the burden on the Logstash agent by receiving documents from many Logstash agents queuing them and then streaming them into a central Logstash agent to be outputted into the elasticsearch cluster.

To evaluate the performance of elasticsearch the scalability of the search server was evaluated first. To do this elasticsearch instances were setup and deployed to ensure that the master would automatically commission and de-commission the nodes without disruption to service. Again these tests were carried out successfully on local hardware. However when tested on ec2 it took some configuration 'tweaks' to get the cluster up and running but once a configuration template was developed no issues were encountered in the deployment in regards to commissioning and decommissioning of nodes. Elasticsearchs performance in terms of writes and reads/searches was also very impressive with instant reachability of documents observed once committed to the cluster thus ensuring the ability to perform real time analytics and search. When Kibana was integrated with elasticsearch queries were made using the search-bar and metrics were investigated. All terms expected were returned. Sample queries were made against the documents in the elasticsearch cluster against log files which might show as a security issue such as failed login attempts and induced system errors returning successful results. Also the metrics which Kibana produces were also investigated such as the average value of memory usage and CPU usage which was cross referenced with log files to try and determine a correlation. Also accuracy of timestamps was investigated to ensure accurate reporting if metrics was provided to ensure document accuracy.

The previous test cases were used to ensure an accurate and reliable dataset was being provided to Rapidminer for analysis. In Rapidminer the support vector machine algorithm was applied to prove how security in a virtual environment can be improved through careful planning , system monitoring and intelligent use of system data. The results from the support vector machine algorithm help prove that the classification of system events can be carried out accurately using the right analysis

techniques and data. The outcome of this will prove vital in helping an administrator identify vulnerabilities in a system. It is then up to the administrator to use the information provided to their advantage.

The support vector machine algorithm was deployed to develop a training model for classification of a dataset with just under twenty thousand records containing information on system activities derived from log files and system usage data retrieved from ganglia. The support vector machine algorithm was applied to the dataset and carried out over one hundred thousand iterations to try and find the optimal classification training model for the data. But in order to evaluate the training model it was validated by applying it to an unlabelled dataset and calculating the accuracy of the classification, in other words how successful the training set was applied.

The validation process involved carrying out one hundred validations of the training model based on a shuffled sampling of the dataset and a linear sampling of the dataset. This means that random samples of unlabelled variables were selected from the dataset and through the support vector machines training model were classified. The accuracy of this classification were defined by the root mean squared error which describes the accuracy of the learner model in comparison to the training model. The lower the root mean squared error the more accurate the prediction. In this case the root mean squared error returned a value of 1.198 +/- 0.162 with a squared error of 1.461 +/- 0.385 which means the model was very successful in classifying the unlabelled dataset.

At first glance, when using linear sampling instead of a shuffled sample more accurate results were achieved through the validation process. Linear sampling returned a root mean squared error of 0.610 +/- 1.069 and a squared error of 1.515 +/- 4.538 but with a higher +/- figure it is obvious that the linear sampling model provides less consistent classification as this shows how the results deviated.

These low error rates in the classification process indicate that by using the support vector machine learning algorithm with shuffled sampling a SVM training model can accurately classify system events into different classes. By carrying out this process more indept analysis of a system and its useage can be carried out in order to improve system security.

Chapter 6

Conclusions

From researching procedures and techniques into improving the security of a virtualised environment it can be concluded that the security of a virtualised environment shares a lot of common characteristics with a traditional information technology environment, however there are some areas which require more attention than a traditional one and some parameters which can cause different opportunities for attack or service disruption in general. The policies which govern a virtualised environment are paramount to the entire systems security and must be monitored and controlled on a constant basis.

To help secure virtualised environments, in-depth analysis along with distributed monitoring tools can be implemented in order to gain a deeper insight into the environment. It is evident from research that a stronger focus by the system administrator on the collection of system metrics and event logs are needed in virtualised environments in order to fully secure them. If not the environment, the implementation of policies and system security auditing can become a very complex process so it must be ensured that system maintenance is never taken for granted and managed efficiently and effectively.

The environment implemented and tested as part of this research demonstrates a way for system administrators to gather, collect and analyse data on a virtualised environment accurately in a distributed manner which can help in maintaining a systems security through organised and streamlined system event log collection and analysis.

It has been proven from the result of the support vector machine learning model that accurate classification of system events can be determined using the support vector

machine learning algorithm. Therefore the model produced can be used to analyse the environment and help identify threats and vulnerabilities in the system.

Future development of this model could be application integration to help automate the analysis process. This would be an ideal use case for a large virtualised environment as it would reduce the amount of administrative tasks needed to be carried out by the administrator. This use case could also supply real time analytics if applied efficiently which in the deployment discussed in this document would be unfeasible due to the manual nature of the analytics process described.

In conclusion it has been demonstrated that with the correct use of tools, accurate selection of metrics and intelligent use of analytical tools that virtualisation can be a more secure environment to deploy applications and provide computational resources.

Chapter 7

Appendix

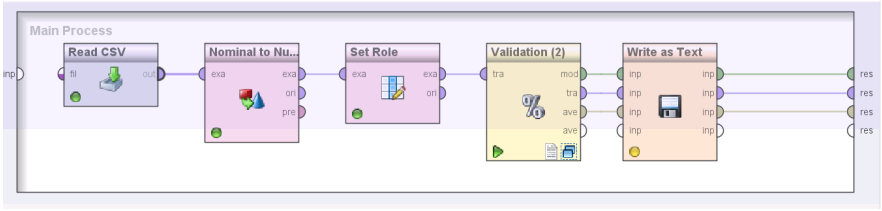

Fig 1: Rapidminer workflow

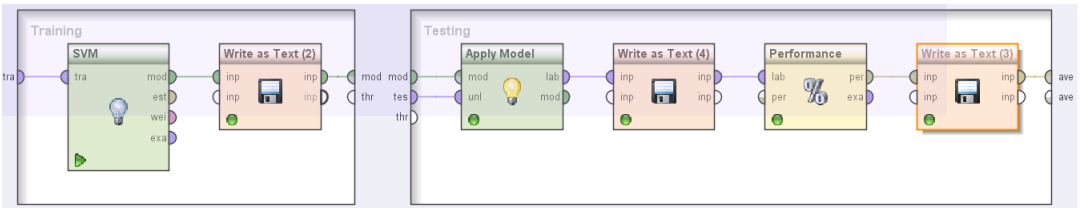

Fig 2: Support vector machine application, application and validation

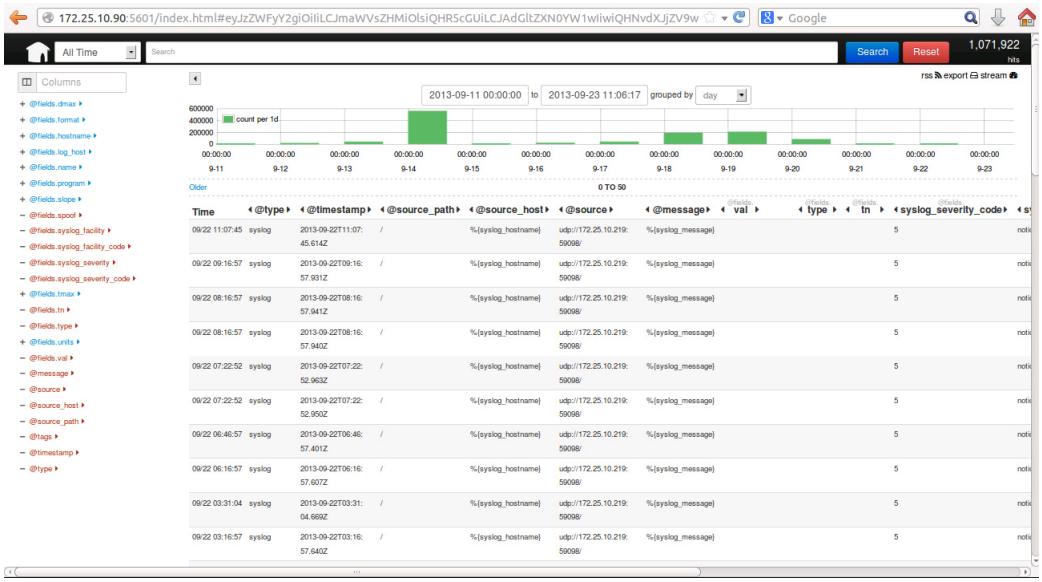

Fig 3: This is the kibana Web user interface

```
globals {
  daemonize = yes
  setuid = yes
  user = ganglia
  debug_level = 0
  max_udp_msg_len = 1472
  mute = no
  deaf = no
  allow_extra_data = yes
  host_dmax = 0 /*secs */
  cleanup_threshold = 300 /*secs */
  gexec = no
  send_metadata_interval = 0 /*secs */
}

/*
 * The cluster attributes specified will be used as part of the <CLUSTER>
 * tag that will wrap all hosts collected by this instance.
 */
cluster {
  name = "eucalyptus"
  owner = "marc_reilly"
  latlong = "cloud competency center"
  url = "@marcyreilly"
}

/* The host section describes attributes of the host, like the location */
host {
  location = "vmware esxi"
}

/* Feel free to specify as many udp_send_channels as you like.  Gmond

udp_send_channel {
  #bind_hostname = yes
  mcast_join = 239.2.11.71
  port = 8649
  ttl = 1
}
udp_send_channel {
host = 192.168.5.35
#mcast_join = 239.2.11.71
port = 8649
#ttl = 1
}

udp_send_channel {
host = 172.25.10.90
port = 8649
#ttl = 1
}



/* You can specify as many udp_recv_channels as you like as well. */
udp_recv_channel {
  mcast_join = 239.2.11.71
  port = 8649
  bind = 239.2.11.71
}

/* You can specify as many tcp_accept_channels as you like to share
   an xml description of the state of the cluster */
tcp_accept_channel {
  port = 8649
}
```

Fig 4: This is the Ganglia Monitoring daemon configuration

```
cluster.name: elasticsearch
node.name: Master
node.master: true
node.data: true

cluster.name: elasticsearch
node.name: slave1
node.master: false
node.data: true
```

Fig 5: Elasticsearch Configuration. The top section is located on the master while the second section is located on the slave

```
input {
ganglia{
type => "ganglia"
}

  udp {
    port => 5000
    type => syslog
  }
}

filter {
  grok {
      type => "syslog"
      pattern => [ "<%{POSINT:syslog_pri}>%{SYSLOGTIMESTAMP:syslog_timestamp} %{SYSLOGHOST:syslog_hostname} %{DATA:syslog_program}(?:\[%{POSINT:syslog_pid}\])?: %{GREEDYDATA:syslog_message}" ]
      add_field => [ "received_at", "%{@timestamp}" ]
      add_field => [ "received_from", "%{@source_host}" ]
  }
  syslog_pri {
      type => "syslog"
  }
  date {
      type => "syslog"
      match => [ "syslog_timestamp", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]
  }
  mutate {
      type => "syslog"
      exclude_tags => "_grokparsefailure"
      replace => [ "@source_host", "%{syslog_hostname}" ]
      replace => [ "@message", "%{syslog_message}" ]
  }
  mutate {
      type => "syslog"
      remove => [ "syslog_hostname", "syslog_message", "syslog_timestamp" ]
  }
}

output {
elasticsearch_http {
        flush_size => 50  # number (optional), default: 100
        host => "127.0.0.1"   # string (optional)
  }
stdout{ }
}
```

Fig 6: Logstash configuration

```
#
# Apache access file:

$ModLoad imfile
$InputFileName /var/log/apache2/access.log
$InputFileTag apache-access:
$InputFileStateFile stat-apache-access
$InputFileSeverity info
$InputRunFileMonitor

#Apache Error file:

$InputFileName /var/log/apache2/error.log
$InputFileTag apache-errors:
$InputFileStateFile stat-apache-error
$InputFileSeverity error
$InputRunFileMonitor

$InputFilePollInterval 10

$ModLoad imfile
$InputFileName /var/log/eucalyptus/*.*
$InputFileTag eucalyptus:
$InputFileStateFile euca-log
$InputFileSeverity info
$InputRunFileMonitor

/var/log/libvirt/*.* @172.25.10.90:5000
/var/log/eucalyptus/*.* @172.25.10.90:5000
*.* @172.25.10.90:5000
```

Fig 7: rSyslog configuration for the Eucalyptus machines

```
# provides support for local system logging
$ModLoad imuxsock

# provides kernel logging support (previously done by rklogd)
$ModLoad imklog

# provides UDP syslog reception. For TCP, load imtcp.
#$ModLoad imudp
# Provides UDP syslog reception
$ModLoad imudp
$UDPServerRun 514
#
*.* @192.168.5.3:5000
*.* @172.25.10.90:5000
```

Fig 8: rSyslog configuration for the central log server

Bibliography

[1] R. Anand, S. Sarswathi, and R. Regan. Security issues in virtualization environment. In *Radar, Communication and Computing (ICRCC), 2012 International Conference on*, pages 254–256, Dec.

[2] R. Anand, S. Sarswathi, and R. Regan. Security issues in virtualization environment. In *Radar, Communication and Computing (ICRCC), 2012 International Conference on*, pages 254–256, Dec.

[3] K.V. Bellur, M. Krupal, P. Jain, and P. Raghavendra. Achieving operational efficiency with cloud based services. In *Computer Science Education (ICCSE), 2011 6th International Conference on*, pages 1063–1068, Aug.

[4] Yanpei Chen, Vern Paxson, and Randy H. Katz. Whats new about cloud computing security? Technical Report UCB/EECS-2010-5, EECS Department, University of California, Berkeley, Jan 2010.

[5] Inc Gartner. Gartner Says Worldwide Cloud Services Market to Surpass 109 Billion dollars in 2012, 2012.

[6] Inc Gartner. I.T. Glossary - Virtualization Definition, 2013.

[7] John Linwood Griffin, Trent Jaeger, Ronald Perez, Reiner Sailer, Leendert Van Doorn, and Ramón Cáceres. Trusted virtual domains: Toward secure distributed services. In *Proceedings of the 1st IEEE Workshop on Hot Topics in System Dependability (HotDep05)*. Citeseer, 2005.

[8] Kai Hwang and Deyi Li. Trusted cloud computing with secure resources and data coloring. *Internet Computing, IEEE*, 14(5):14–22, Sept.-Oct.

[9] Diana Kelley and Security Curve. How data-centric protection increases security in cloud and virtualization. Technical report, Cloud Security Alliance, Nov 2011.

[10] Elliot Lewis. Threat modeling the landscape of virtualization. Technical report, Dell inc., Aug 2012.

[11] F. Sabahi. Virtualization-level security in cloud computing. In *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, pages 250–254, May.

[12] Karen Scarfone, Murugiah Souppaya, and Paul Hoffman. Guide to security for full virtualization technologies. *NIST Special Publication*, 800:125, 2011.

[13] A.A. Semnanian, J. Pham, B. Englert, and Xiaolong Wu. Virtualization technology and its impact on computer hardware architecture. In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, pages 719–724, April.

[14] Dave Shackleford. The Virtualization Security Landscape: What's Changed?, 2012.

[15] Juraj Somorovsky, Mario Heiderich, Meiko Jensen, Jörg Schwenk, Nils Gruschka, and Luigi Lo Iacono. All your clouds are belong to us: security analysis of cloud management interfaces. In

Proceedings of the 3rd ACM workshop on Cloud computing security workshop, CCSW '11, pages 3–14, New York, NY, USA, 2011. ACM.

[16] Hsin-Yi Tsai, M. Siebenhaar, A. Miede, Yu-Lun Huang, and R. Steinmetz. Threat as a service?: Virtualization's impact on cloud security. *IT Professional*, 14(1):32–37, Jan.-Feb.

[17] U. Tupakula and V. Varadharajan. Tvdsec: Trusted virtual domain security. In *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, pages 57–64, Dec.

[18] Hitoshi Ueno, Satomi Hasegawa, and Tomohide Hasegawa. Virtage: Server virtualization with hardware transparency. In *Euro-Par 2009–Parallel Processing Workshops*, pages 404–413. Springer, 2010.