Malware Detection in the Context of Cloud Computing

M. R. Watson
InfoLab21
Lancaster University
Lancaster, UK
m.watson1@lancaster.ac.uk

***Abstract*—Cloud computing is becoming an increasingly popular paradigm due to new services and increased media attention. This increase in popularity has lead to concerns over the security of the cloud, especially from threats such as malware. As more services migrate to a cloud architecture the cloud will become a more appealing target for cyber criminals. In this paper we present the current threats to cloud computing as well as summarising the currently available detection systems for malware in the cloud. We then present a distributed detection system that is aimed at combating the spread of malware in a multi-server cloud.**

***Index Terms*—Malware, Security, Cloud Computing.**

I. Introduction

Recent product releases, such as Apple's iCloud[1], and established products, such as Dropbox[2], have proven that remote storage and seamless access to data across multiple devices are popular features among consumers. In the future we will see an increase in the reliance of cloud computing as more and more consumers move to mobile platforms for their computing needs.

Cloud technologies are made possible through the use of virtualisation in order to share physical server resources between multiple virtual machines (VMs). The advantages of this approach include an increase in the number of clients that can be serviced per physical server and the ability to provide infrastructure as a service (IaaS). Disadvantages include a more complex software stack and a relatively small understanding of security issues.

The security issues of regular operating systems (OSs) are well known due to decades of testing and experience in this area. Security breaches now commonly occur at the application level and are less commonly due to a flaw in the OS itself. Exceptions to this are usually due to the inclusion of new features into an OS kernel either to provide new functionality or to support new hardware. Virtualisation is not only subject to the security issues of applications and operating systems, but also introduces new security issues that are not as well understood, such as the sharing of hardware resources between VMs.

Clouds themselves are composed of a number of virtualised environments which are networked together. The compact topology of this network and the high probability of relative homogeneity across VMs creates an ideal environment for rapid malware propagation. Protecting against malware in the cloud therefore requires a certain level of coordination between virtualised environments if threats are to be reliably detected and dealt with.

In this paper we review previous work on malware detection, both conventional and in the presence of virtualisation in order to determine the best approach for detection in the cloud. We also argue the benefits of distributing detection throughout the cloud and present a novel approach to coordinating detection across the cloud. Section II provides background to the research area, specifically: cloud technologies, security in the cloud, malware detection and detection in the cloud. Section III focusses on malware detection at the hypervisor level and introduces our work in this area. Finally, section IV summarises the points raised in this paper and provides some ideas for future work.

II. Background

A. Cloud Technologies

Cloud computing is an umbrella term for services that offer off-site computing and storage. There are three main types of cloud computing: software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS). Not all of these require virtualisation, SaaS for example could be implemented as a typical client/server service, but virtualisation allows hardware to be better utilised and enables the infrastructure itself to be hired out, as is the case in IaaS.

There are various implementations of virtualisation, but they are all built on the concept of virtual machines (VMs). The VMs exist as a virtual computer system and each have their own operating system (OS) and applications. The VMs are managed by a virtual machine monitor (VMM), which is sometimes referred to as a hypervisor. Commercial virtualisation software exists, such as VMWare ESXi[3], and open source solutions are also available, such as the Xen hypervisor[4] or the Linux KVM hypervisor module[5]. Xen is particularly popular and is used by Amazon for their E2C products[6]. KVM is used by Red Hat in their virtualisation products[7].

The virtualisation products above are all examples of what is known as bare-metal hypervisors. There is also another form of VMM that exists at the application level within an OS; these are known as hosted hypervisors. Each are also commonly referred to as type 1 and type 2 hypervisors respectively. Type 1 hypervisors will be the focus of this research due to their use in servers and cloud services.

The cloud itself is usually composed of many physical server machines, or hardware nodes. These nodes each have their own VMM hosting some VMs. There are a number of reasons for having multiple hardware nodes, the first being limited resources. It is important not to run too many VMs on a single hardware node because of the limited size of RAM and disk space available. With more than one physical machine it is possible to load balance based on CPU, RAM or network utilisation. Another reason for multiple hardware nodes is redundancy. If a fault is detected on a server the VMs can be migrated to another server before it goes down. This is achieved in the same way as load balancing, but solves a slightly different problem.

Access to the cloud is via a network connection and management is usually achieved through a dashboard type interface. Most software designed to interface with the cloud, including management dashboards, are implemented using web technologies. Kaavo[8] is an example of a management interface that can be accessed using a web browser.

B. Security in the Cloud

Previous work on cloud security has suggested that there are a number of security issues associated with cloud computing. In [9]

ISBN: 978-1-902560-26-7

Archer et al. provide the following threats to cloud computing:

- Threat 1: *Abuse and Nefarious Use of Cloud Computing*
- Threat 2: *Insecure Interfaces and APIs*
- Threat 3: *Malicious Insiders*
- Threat 4: *Shared Technology Issues*
- Threat 5: *Data Loss or Leakage*
- Threat 6: *Account or Service Hijacking*
- Threat 7: *Unknown Security Profile*

Of these, threats 2 and 4 are directly related to malware. *Insecure Interfaces and APIs* would allow malware running on one VM to execute code or access data on another VM. *Shared Technology Issues* include the sharing of physical memory between multiple VMs. This could lead to a new form of worm that, instead of spreading via networks, could spread by writing to the memory owned by another VM. This kind of propagation would be unique to virtualised environments.

Threats 1, 3 and 6 could be indirectly related to malware, for example in the deployment of malware by malicious individuals. *Abuse and Nefarious Use of Cloud Computing* is possible due to the relative anonymity of cloud subscription. Malicious organisations could use cloud space as a platform to launch attacks. *Malicious Insiders* is a similar threat, but instead of being customers the malicious individuals are instead employees of the cloud providers. *Account Hijacking* is a common threat throughout the Internet and would allow malicious individuals to perform similar actions to threats 1 and 3.

Threats 5 and 7 are not related to malware and are concerned with data loss, which is a natural occurance in computer systems, and the opacity that is inherent in the cloud. *Data Loss or Leakage* is exacerbated in virtualised environments because the system as a whole is more complicated than a single OS computer system. *Unknown Security Profile* is in contrast to in-house servers where the implementation of data storage and networking is known. A customer has no guarantee that the security measures promised by a cloud provider are actually in place; there is a level of opacity that is not an issue in alternatives to the cloud.

In [10] Payne identifies similar problems with virtualisation including an increase in software complexity, therefore increasing the opportunity for vulnerabilities; and hardware sharing, which causes computer systems that used to be separated by an air gap to be separated solely by software. When using software, especially complex software, there is always a risk of an improper implementation or configuration, more so than when using hardware for the same task. Take for example a simple server. The remotely exploitable vulnerabilities are confined to the OS and application software. The same server implemented as a VM is subject to the vulnerabilities of the VMM, OS and applications. It can therefore be assumed that hardware sharing under the management of software is inherently less secure than distinctly separate machines.

C. Malware Detection

Malware detection has been an important issue in computing since the late ’80s. Since then the predominant method of malware detection is to scan a computer system for infection by matching malware signatures to files on the computer. Although detection of known samples is extremely reliable, signature based detection only works for malware that has been obtained, analysed and a suitable signature identified. In [11] Murad et al. show that signature based detection can be thwarted by analysing the malware instructions and identifying the instructions that comprise the signature. By altering this specific portion of code it is possible to evade detection; in effect the process takes a known sample and converts it into an unknown sample.

Another downside to signature based detection is the maintenance of the signature database. With the constant evolution of malware and the polymorphic nature of many samples it has become necessary to drop old samples from databases. If this practice continues malware samples which have already been identified will become undetectable and will once again become useful to cyber criminals.

Other malware detection techniques are available in order to overcome the problems of obfuscation and polymorphism. Instead of scanning for matching signatures it is possible to analyse the behaviour of a malware sample and base detection on observation of running processes. There are a variety of ways this could be achieved. One approach is to monitor the process names themselves. Unfamiliar or uncommon processes can be assumed to be malicious until further information can be obtained[12]. Another approach is to base detection on the behaviour of the process itself. If a process begins executing instructions that match the behaviour of a known malware sample then that process can be considered harmful. Similar techniques can be applied to the monitoring of network activity. If certain addresses or port numbers, or some other features, are present in the traffic directed towards or away from the computer system it can be assumed that malware is either targeting the system or is already running within the system.

The downside to both signature and behaviour-based detection is that they occur within the OS itself. This gives malware the opportunity to alter the information that is provided to the detection software by the OS. If, for example, the security software polls the OS for a list of running processes it is possible that malware can alter this list so that the malware process itself is not present in the list. The detection software will then have no knowledge of the malware process and the malware will have escaped detection. This behaviour is usually associated with rootkits, but could be employed by any malware.

To combat this Kaspersky offers a sandboxing product called Safe Run[13]. Non-OS processes can be encapsulated in a safe execution environment that monitors for malware using both behaviour-based and signature-based detection. There is, however, no guarantee that malware has not infected the OS prior to installation of the detection software, or that infection could occur due to processes running outside of the sandbox. As long as the detection software exists in the same execution environment as other processes, including malware processes, there is an opportunity for subversion. A better approach would be to perform the detection from outside looking in.

D. Detection in the Cloud

As mentioned in the previous subsection, detection would be best achieved from outside of the OS. This is possible in clouds because they are built on virtualisation which encapsulates each OS in its own VM. Detection is now possible by executing detection software in a privileged domain within the virtualisation environment. Jiang et al. advocate this approach to malware detection in [14] and address the challenge posed by the semantic gap between the VMM and its VMs. Payne builds on this work by further developing this technique and providing libraries that make it easier for developers to create monitoring software through VM introspection[10]. The original library, named XenAccess, has since been superseded by the LibVMI library[15], also developed by Payne.

Detection in the cloud not only enables introspection, it could also improve the reliability of statistics based approaches. Behavioural and anomaly detection techniques are built on statistical analysis

and are subject to a level of uncertainty. In an isolated computer system this uncertainty cannot be improved upon because access to additional information is not possible. Detection at the hypervisor level, however, can combine the data from many VMs which has the potential to reduce uncertainty and false positives in any results.

Although the solutions for malware detection discussed so far seem promising there is another security risk that is unique to the cloud. The compact network topology of clouds coupled with the likelihood of homogeneous software deployment could allow rapidly propagating malware, such as worms, to propagate even faster and with an increased success rate. This indicates that coordination across the cloud is an important consideration. Not only would a certain level of communication between detection software

III. VMM Based Detection

Virtualisation offers a number of advantages over a conventional single OS computer system when it comes to malware detection. Assuming the VMM isn't compromised, detecting malware from the VMM provides an outside-in perspective, which cannot be modified by malware. In-OS detection occurs at the application level and is dependent on the OS for information. Malware that is operating at the kernel level, such as a rootkit, is able to alter the information provided by the OS and thus mask any signs of malware. Malware can also turn off detection systems altogether.

Another advantage of detection outside the OS is the ability to combine various types of data that have been gathered from outside the OS. Again, we have to assume the VMM isn't compromised, but if this is the case the view of the VM from the outside is representative of the actual state of the machine at any given moment. It is therefore possible to obtain and combine a variety of data which would lead to the detection of malware. The first is the state of memory which can be obtained through memory introspection. The second is access to the file system by monitoring the disk image. The third is access to a baseline disk image which can be obtained by copying the VM's disk image before booting the VM. Finally, the fourth is network activity which is available through the monitoring of any network bridges which connect VMs.

A. Implementing Detection

The platform that has been chosen for the development of VMM-based malware detection systems is the Xen hypervisor. This is due to its open source licensing and wide spread adoption. If time permits, any systems developed for Xen will also be adapted to run on KVM. The purpose of this is to determine which aspects of the system can be generalised and which are tied to a particular hypervisor architecture.

VM monitoring will be achieved through the use of pre-existing libraries. An example of one such library, LibVMI, was given in section III.

There is already a large amount of prior work on the technique of VM monitoring. We hope to take this work further by using VM introspection to coordinate detection in the cloud. Work has so far focussed on detection within a single hypervisor, but we argue that this approach is only appropriate for a minority of cloud implementations which use only a single physical server. Malware detection and remediation could be automated in the cloud, but only if there is a level of collaboration between the detection systems that are deployed throughout the cloud.

B. Distributed Detection

One of the main issues with monitoring VMs in a cloud is the possibility of the cloud being composed of many physical servers. For security of the cloud to be assured the cloud as a whole should be monitored. For this to be achieved there would need to be some form of communication between individual detection instances. We therefore present a novel detection system to be deployed at the virtualisation level of a cloud which is composed of a number of collaborative agents. Previous research in this area, such as that carried out by Biedermann et al. in [12], gather data at each hypervisor, but then carry out the detection from a single point in the entire architecture. Our approach is to gather information and perform analysis at each hypervisor in order to distribute points of failure. In a system with a single point of detection the traffic to a single point would be an issue. Traffic to one point has the potential to congest network links and by distributing the detection it would reduce the load on a single point while at the same time eliminating the single point of failure which would otherwise be present.

The system we propose is built on the concept of individual detection agents, each deployed inside the VMM of a physical machine in the cloud. Each agent will use a combination of detection techniques, in particular making use of network traces, obtaining VM memory state information and accessing the VM disk image for filesystem data. This will be achieved through the use of an appropriate introspection library, such as LibVMI. A diagram of our proposed system is shown in figure 1. The introspection is indicated by the lines that extend from agent 1 to the various VMs hosted in server 1. The VMs in server 2 are not shown in order to simplify the diagram.

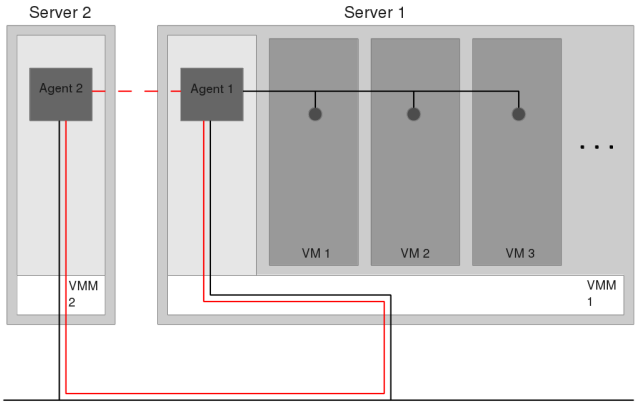

Fig. 1. A Malware Detection System Based on Agents

Since a single cloud can be made up of multiple physical machines each agent should not operate as an individual, but as part of a network of agents. This will be made possible through the development of a secure protocol that will allow information to be shared between agents in the same cloud. By sharing information on the overall health of the cloud it will enable the agents to determine the threat level to the VMM or VMs with which they are associated. In figure 1 this secure communication is indicated by the dashed line between agents 1 and 2. The actual route for the traffic is via the physical connection between the two servers.

C. Agent Architecture

The architecture of the agents is purely conceptual at this stage, but there are a number of features which need to be incorporated into the design of each agent. The first is a method for gathering data from each of the VMs that belong to a single VMM. The second is an algorithm for analysing the data and determining the presence or

absence of malware in a particular VM. The third is a mechanism for updating each VM on a single machine with the current threat level. Finally, the fourth is a protocol for communicating with other agents in the cloud.

The method of gathering data has already been discussed and will primarily be achieved through VM introspection.

In choosing an algorithm for data analysis we would prefer to avoid signature based detection due to its inadequacy in detecting new or modified strains of malware. It would therefore be advantageous to employ some form of heuristic analysis in the agents, possibly incorporating some form of machine learning. It is anticipated that malware which makes use of features unique to virtualisation will be undetectable through signature based methods for long enough to do significant damage. Detection based on the behaviour of the malware or the detection of anomalies in VMs will have a better chance of detecting modern propagation mechanisms. As previously mentioned in subsection II-D, behavioural or anomaly detection algorithms would also benefit from a distributed and collaborative system as opposed to detection confined to a single node. This is another reason for our choice of collaborative agents.

Updating each VM is an essential feature. The purpose of coordinating detection throughout the cloud is to automate as much as possible the malware detection and remediation process. Without a VM notification mechanism it is impossible to perform a remediation phase if infection is detected. It is also impossible to increase any defences if threat levels increase. This feature will be largely dependent on the hypervisor architecture and will leverage some of the tools already available to administrators of a particular platform. Xen, for example, provides *xm* as a management user interface tool.

Distribution, collaboration and automation are the main themes of this research, communication will therefore form an essential part of each agent. The distributed nature of the detection system requires a decentralised peer-to-peer (P2P) architecture. This is because a centralised system would be vulnerable at the single point where data is disseminated. There could also potentially be congestion of the network by forcing all information to a single point and back out again.

It is not known whether the development of an overlay network protocol is necessary, since there are already P2P protocols available, but any protocol will need to be secure and timely. An insecure protocol would open up possibilities for malicious parties to disrupt communication and alter inter-agent communication, which would render the system useless. An untimely protocol would introduce lag into the system, which would allow malware to spread faster than the system can react.

The internal architecture of a single agent in the system is shown in figure 2. The functionality of the agent is split into four modules, each responsible for one of the four essential behaviours of the agent.

Control will predominantly reside in the data analysis module, but will be passed to the other modules when necessary. The data analysis module will instruct the introspection module to gather data from each of the VMs that are hosted by the VMM. The introspection module will gather and store data as well as querying its feature database in order to satisfy the request from the data analysis module. The data analysis module will then use its inbuilt malware detection algorithms to analyse the data and determine current threat levels based on data from the VMs as well as data that has been communicated from other agents in the system. Any conclusions drawn from data analysis will be passed to the overlay network module so that they can be communicated to other agents. If threat levels for a particular VM are high a request will be sent to the VM notification module in order to enact an enhanced defence phase or remediation phase in the VM.

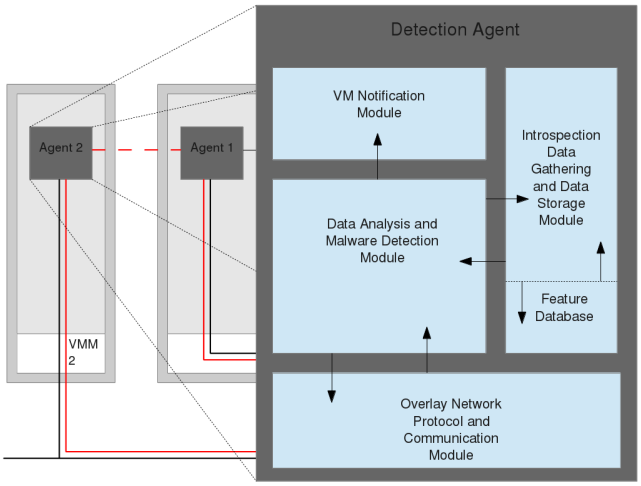

Fig. 2. Internal Architecture of a Detection Agent

IV. Conclusion

In this paper we summarised the security issues facing cloud computing. It was determined that as well as conventional attack vectors, which are present in operating systems, virtualisation also introduces new opportunities for malware writers. These are due to the sharing of physical resources through software mechanisms, which if implemented incorrectly would allow malware to access the memory in other VMs. This could lead to new forms of malware that spread in a worm-like way by writing to memory instead of spreading via the network.

In addition to the potential for new attack vectors it was also concluded that the distributed nature of multi-server clouds leaves clouds vulnerable to attack from the inside by conventional malware. A worm would be able to spread from VM to VM quickly because of the compact topology of cloud networks.

In subsection III-B we presented a new method of detection based on a distributed network of detection agents in order to coordinate detection across the whole cloud. This would provide a mechanism for locking down vulnerable VMs in the event that malware is detected in the cloud. A single agent system would only be effective in securing VMs on a single physical server. A system built on multiple points of detection, but consolidated to a single point is more vulnerable to attack due to the single point of failure.

A. Future Work

Future work in this area will focus on the development of detection systems based on memory introspection and heuristic or statistical detection, as opposed to signature-based detection. It will also focus on the reliability that is gained from data sharing as opposed to gathering and analysing data from a single computer system only.

There is also the possibility of exploring new attack vectors on clouds and determining how feasible it would be to create a ‘memory worm’. This is an interesting concept, but is not really in the scope of this research. It will be explored if our other goals have been met.

As well as establishing the possibility of new attack vectors it will also be important to determine the actual threat conventional worms

pose to the cloud. In particular we will analyse the behaviour of existing worms and botnet software in virtualised environments.

The peer-to-peer nature of the proposed detection system will require a secure protocol in order to communicate information across the cloud. This will be the focus of research once the requirements of detection are known. It is important to first determine how much of an improvement to detection can be gained from data exchange between agents and which algorithms will benefit most from this.

ACKNOWLEDGEMENT

Funding for this research is from the EPSRC with additional funding from BT. My thanks go in particular to Ben Azvine and Fadi El-Moussa at BT; Andreas Mauthe, Alberto Schaeffer-Filho and David Hutchison at Lancaster; and Syed Shirazi who has recently joined us at Lancaster.

REFERENCES

[1] Apple Inc., "Apple iCloud." icloud.com webpage, https://www.icloud.com/ (accessed 13/04/12).

[2] Dropbox, Inc., "Dropbox." dropbox.com webpage, https://www.dropbox.com/ (accessed 13/04/12).

[3] VMware, Inc., "VMware ESXi." vmware.com webpage, http://www.vmware.com/products/vsphere-hypervisor/overview.html (accessed 13/04/12).

[4] Citrix Systems, Inc., "Xen." xen.org webpage, http://www.xen.org/ (accessed 13/04/12).

[5] Red Hat, Inc., "KVM." linux-kvm.org webpage, http://www.linux-kvm.org/page/Main_Page (accessed 13/04/12).

[6] Amazon, "Amazon Web Services: Overview of Security Processes." aws.amazon.com white paper, aws.amazon.com/articles/1697, 2008.

[7] Red Hat, "Red Hat Enterprise Virtualization for Servers." redhat.com datasheet http://www.redhat.com/promo/rhev3/pdf/rhev_for_servers_datasheet.pdf.

[8] Kaavo, "Kaavo Cloud Management Software." kaavo.com webpage, http://www.kaavo.com/ (accessed 30/05/12).

[9] J. Archer, D. Cullinane, N. Puhlmann, A. Boehme, P. Kurtz, and K. Reavis, "Top Threats to Cloud Computing." Cloud Security Alliance white paper, https://cloudsecurityalliance.org/research/initiatives/top-threats/, 2010.

[10] B. D. Payne, *Improving Host-Based Computer Security Using Secure Active Monitoring and Memory Analysis*. PhD thesis, Georgia Institute of Technology, USA, 2010.

[11] K. Murad, S. Shirazi, Y. Zikria, and I. Nassar, "Evading Virus Detection Using Code Obfuscation," in *Future Generation Information Technology*, vol. 6485 of *Lecture Notes in Computer Science*, pp. 394–401, Springer Berlin / Heidelberg, 2010.

[12] S. Biedermann and S. Katzenbeisser, "Detecting computer worms in the cloud," in *Proceedings of the 2011 IFIP WG 11.4 international conference on Open Problems in Network Security*, iNetSec'11, (Berlin, Heidelberg), pp. 43–54, Springer-Verlag, 2012.

[13] K. Lab, "Safe Run Technology in Kaspersky Internet Security 2011." kaspersky.com whitepaper http://www.kaspersky.com/images/sandbox_whitepaper-10-75831.pdf.

[14] X. Jiang, X. Wang, and D. Xu, "Stealthy Malware Detection Through VMM-based "Out-of-the-Box" Semantic View Reconstruction," in *CCS '07 Proceedings of the 14th ACM Conference on Computer and Communications Security*, 2007.

[15] B. D. Payne, "LibVMI Introduction." vmitools Google Code project page, http://code.google.com/p/vmitools/wiki/LibVMIIntroduction (accessed 09/04/12).