Vulnerability Analysis using Network Timestamps in Full Virtualization Virtual Machine

M. Noorafiza[1,2], H. Maeda[1], R. Uda[1], T. Kinoshita[1] and M. Shiratori[1]
[1]*Graduate School of Computer Science, Tokyo University of Technology, 1404-1 Katakuramachi, Hachioji Tokyo, Japan*
[2]*Department of Computer Science, National Defence University of Malaysia, Kem Sungai Besi, Kuala Lumpur, Malaysia*

Keywords: Vulnerability, Security, Privacy, Malware, Virtual Machine, Cloud Computing.

Abstract: Virtualization is the main underlying technology for cloud computing. The popularity of cloud computing had expanded rapidly over the past few years. As any new technology advancement, cloud computing also has vulnerability possibilities and potential security risks. Therefore it is important to study and understand the underlying technologies in cloud computing and test any possible loophole that may give advantages for malware and attackers. Virtual machine (VM) is one of the basic component in cloud computing. VM itself is a program that executes multiple operating systems on one physical machine. Due to the complexity of the VM, together with the complex setting of the network environment and physical machine technology during the implementation of VM environment, vulnerability in the environment may occur. For example, the ability of malware to detect either the environment that they are attacking is on VM or not. Through this detection, the malware or attackers may hide its malicious program since VM are commonly used as defensive system for malware detection, such as honeypots. In this paper, we present a remote detection technique for VM that uses IP timestamp option in full virtualization that could be used to detect VM environment and contributing to VM vulnerability. Evaluation of this technique was done by examining and analysing the characteristic of IP packet timestamps replies from VM and real machine. This research finding could serve as new knowledge for further studies on how to provide comprehensive protection from VM vulnerability. This research also could formulate more effective security improvement that could lead to better security policy towards VM technology.

1 INTRODUCTION

Virtualization technologies allow multiple operating systems and applications run on the same machine that resulted on effective time and cost for deployment. Virtualization is also one of the important pillar for cloud computing. Through cloud computing, users will be able to conveniently access the applications and services on the cloud via their thin client or mobile devices. The user friendly characteristic makes the popularity of cloud computing sky rocketing. (Buyya, R., et al., 2009) argues that cloud computing are expanding until cloud computing itself could be considered as the next vital utility after water and electricity. However, virtualization technology may also serve as the limitation of cloud computing since there are potential security risks that associated with VM technology (Anthes, G., 2010). This limitation may causes enterprises and organizations to hold back from implementing cloud computing as their IT solution. Such example for the vulnerability and security issues in VM is the ability of malicious code to detect the presence of VM (Grobauer, B., et al., 2011). Despite of enormous effort in keeping the differences of running on VM and real machine environment as minimum as possible, there are still possible methods in detecting the virtual machine environment.

In this paper, we present a remote detection technique for VM that uses IP timestamp option in full virtualization that could be used to detect VM environment and contributing to VM vulnerability. IP timestamp option allows a requester to request timestamps value from any machine which handles packets by specifying its IP address. This research is the continuity from our previous work and we are now focusing on testing VM remote detection method using network timestamps in full virtualization. Our previous study (Noorafiza, M., et

Noorafiza M., Maeda H., Uda R., Kinoshita T. and Shiratori M..
Vulnerability Analysis using Network Timestamps in Full Virtualization Virtual Machine.
DOI: 10.5220/0005242000830089
In *Proceedings of the 1st International Conference on Information Systems Security and Privacy* (ICISSP-2015), pages 83-89
ISBN: 978-989-758-081-9
Copyright © 2015 SCITEPRESS (Science and Technology Publications, Lda.)

al., 2013) proved that there were distinguishable differences in the timestamps replies behaviours received from para-virtualization VM and non-VM machines even in a high performance private cloud computing environment. Findings from this paper proved that there were distinguishable differences in full virtualization as well. This research is vital in contributing to comprehensive analysis for network timestamps pattern that could lead to detection of VM that was implemented using full virtualization technique and the possible attack vector. This finding also provide more input for researchers in initiating and establishing security improvement and also formulating more effective and better security policy towards VM technology in full virtualization.

2 RESEARCH BACKGROUND

2.1 Virtualization and Cloud Computing

Virtualization technology is the key feature for cloud computing. A host computer runs software program known as a hypervisor or Virtual Machine Monitor (VMM) that creates one or more VM that simulate physical computers. There are two types of hypervisor, para-virtualization and full virtualization. Differences of characteristic between the two types of virtualization are shown in Figure 1. Full virtualization offers the best isolation and security for virtual machines. It simplifies migration and portability as the same virtualized guest OS instance can run on native hardware. In full virtualization, the hypervisor provides most of the same hardware interfaces as those provided by the hardware's physical platform. In other word, the OSs and applications running within full virtualization do not need to be modified for virtualization to work if the OSs and applications are compatible with the underlying hardware. For full virtualization, as the guest OS is fully abstracted from the underlying hardware by the virtualization layer, the guest OS does not aware it is being virtualized and requires no modification, therefore making the implementation simpler. In addition, full virtualization is the only option that requires no hardware system assist to virtualize sensitive and privileged instructions.

The hypervisor translates all operating system instructions and caches the results for future use, while user level instructions run unmodified at native speed. In full virtualization technique, speed of executing the instructions will be slower than speed of executing instructions in para-virtualization

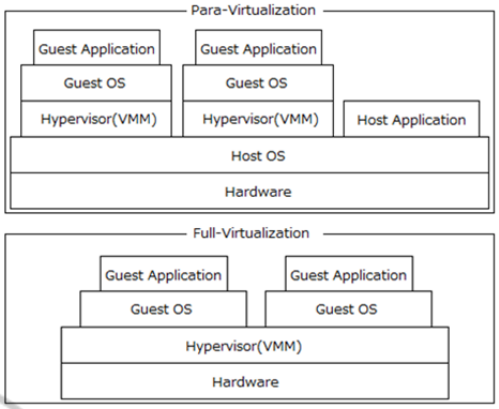

Figure 1: Two Types of Virtualization,

because hypervisor in full virtualization run similar to physical machine that will cause some delay that are able to be detected by attackers and malicious software. Meanwhile, para-virtualization is a method for the hypervisor to offer hardware interfaces to the guest OS to use instead of using normal hardware interfaces. Para-virtualization is lower in virtualization overhead, but the performance advantage of para-virtualization over full virtualization can vary greatly depending on the workload. As para-virtualization cannot support unmodified operating systems, its compatibility and portability is poor. Para-virtualization can also introduce significant support and maintainability issues in production environments as it requires deep OS kernel modifications (VMWare, 2007).

2.2 Security Issues

Despite of the advantages of virtual machine technology and cloud computing, security issues remain as the main issue. One of the biggest challenges in cloud computing is the design of the VM itself that could lead to various security vulnerabilities and threats. Meanwhile, details of the underlying resource architectures where these systems operate are not commonly published for research. Security issues in virtualization technologies remain exist and this scenario is contributing to the vulnerability of cloud computing. Hypervisor technologies and implementation technique of VM also varies. Each scenario of VM implementation both in full virtualization and para-virtualization should be tested and analysis should be made. However, with the fast track changes in the VM technologies, more comprehensive and fast track study need to be performed periodically. Making virtual and native hardware indistinguishable, thus preventing VM detection is vital in order to

minimize security issues towards VM. Therefore, security risk towards cloud computing that utilize VM as the main technology could be reduced.

By VM detection, malicious system could withdraw any harmful operations such as botnet attack and hiding itself from the VM security systems. As the result, malware may avoid from being detected by VM security applications, thus reducing the risk for their behavior from being studied and revealed. Attackers may now write programs that first try to detect either the system are running on VM or not before executing any destructive or security breaching operations. The malware could than selectively targeting to only execute their operation on native machines or client devices such as smart phones and mobile devices. This will creates critical vulnerability in cloud computing. Furthermore if majority of future malware detection such as honeypot runs on virtual machine, malware will eventually choose not to run at all on those environments. The malware attacks will be escaping from detection and exploiting of the VM itself (Ferrie, P., 2007).

Enterprises are also trending in using smart and mobile device that runs on Android, Apple iOS, Apple Mac OS X, Blackberry and etc. This trend is the result from the emerging use of cloud computing environment as information are now easily can be accessed through the cloud computing. Enterprises will provide the mobile devices to their employees in order to give better mobility in completing their daily task. In such cases the required thin client software applications such as those that are related to sales, finance and customer managements will be made available to be downloaded to the devices. However, before the applications could be released to the employees, we predict that there are high possibilities that implementation and testing process for the applications will be done using emulator in the VM on the cloud computing environment. The applications test results might not give the true results, especially in term of security testing against various malicious code because the malicious operation may not show their behavior when they had detected that the running environment are VM. As a result once the application released, the mobile device and other stand-alone environment might be compromised in such a way that the malware will start to execute malicious behavior once it had detected that it is not on a VM environment. Therefore data that are stored or communications through the mobile devices might be revealed to malicious third party.

3 RELATED WORKS

In previous researches, one of the methods for detecting execution within a VM, have typically focused on specific artifacts of the implementation, such as hardware naming, guest-to-host communications systems, or memory addresses. Functional and transparency detection method was discussed in (Ferrie, P., 2007; Garfinkel, T., et al., 2007) by highlighting detection strategies that look upon the characteristic of logical discrepancies, resource discrepancies and timing discrepancies between VM and non-VM environment. Detection method that focuses on differences in performance between VM and physical hardware were also discussed. However, as machines that are being used to host the VM are continuously improved, the difference according to performance might be different and more tests need to be done constantly to verify current situation. A light weight detection method of Virtual Machine Monitor using CPU instruction execution performance stability had been studied in (K. Miyamoto, et al., 2011). However, this method required adjustment to be made in operating system (OS) and could lead to instability in the OS itself. On the other hand, detection method that focuses on network implementation and VM behavior could be considered as a technique for remotely detecting VM without compromising the target. Method that using network timestamps was first exploited by (Kohno, T., et al.,2005) using TCP timestamps as a convert channel to reveal a target host's physical clock skew, which uniquely identifies a physical machine.

Malware will try to avoid honeypots that are mainly implemented in VM to trace and record their behavior and signature. One of the honeypot tools is the automated solution, dynamic malware testing systems TTAnalyze (Bayer, U., et al., 2006) was proposed and became the ideal tool for quickly getting an understanding of the behavior of an unknown malware. This tool automatically loads the sample of malicious code to be analyzed into a virtual machine environment and execute it. The tools recorded the interaction with the operating system that involves recording which system calls were invoked, together with their parameters. This tool could be considered as the early stage of implementation of honeypots in VM. Meanwhile, Temporal Search is a behavior based analysis technique that exploits the fact that, using processor performance to measure time can be inaccurate and the only way for malware to coordinate malicious

events based on time is to use the system's timekeeping infrastructure (Crandall, J. R., et al., 2006). Virtual machine usage in discovering the system timers without making assumptions about the integrity of the kernel was shown. Fuzzy benchmarking was discussed as an approach that can successfully detect the presence or absence of a VMM on a remote system by making timing measurements of the execution time of particular code sequences executing on the remote system (Franklin, J., et al., 2008). Time measurement could be used in order to detect the VM environment. In this research we are focusing on timestamps. Timestamps is the current time of an event that is recorded by a computer. Through mechanisms such as the Network Time Protocol (NTP), a computer maintains accurate current time, calibrated to minute fractions of a second. Network timestamps exist in various network protocols such as Internet Protocol (IP) and Internet Control Message Protocol (ICMP). The IP is designed for use in interconnected systems of packet-switched computer communication networks. It provides info for transmitting blocks of data called data-grams from sources to destinations, where sources and destinations identified by fixed length addresses. IP timestamp (Mills, 1992) is an optional extension to the IP header. It allows the sender to request timestamp values from any machine which handles the packet by specifying it's IP address. However, length of the IP timestamps reply will only limited to nearest microsecond. In this research, for comparison purposes with para-vitualization analysis that we already performed in (Noorafiza, M., et al., 2013). For the VM technologies, we chose the popular virtualization products for most open platforms over the past 5 years which are VMWare ESX (P. Padala, et al., 2007) and Oracle VirtualBox (J. Watson, 2008). Both of these products also could be implemented in full virtualization technique and suitable for our desired experiments environment.

4 EXPERIMENTAL ENVIRONMENTS

In this study, we present a remote detection technique for VM that uses IP timestamp option in full virtualization. To evaluate our technique, experiments were done to obtain IP timestamps reply data to prove that VM is detectable using timestamps discrepancies in full virtualization. The experiments environment were a private cloud computing environment that consist of full-virtualization VM and was set up in our campus lab. Experiments were done in the lab network environment to minimize interruption to the IP timestamps data due to router clock skew. We set up different test environment with high performance machine as a host of 30 running VMs in order to provide maximum utilization for the machine in full virtualization technique. The experiments environment used in this research is shown in Figure 2. In the experiments, packets were sent from stand-alone host OS to guest OS in VM and host in stand-alone environment.

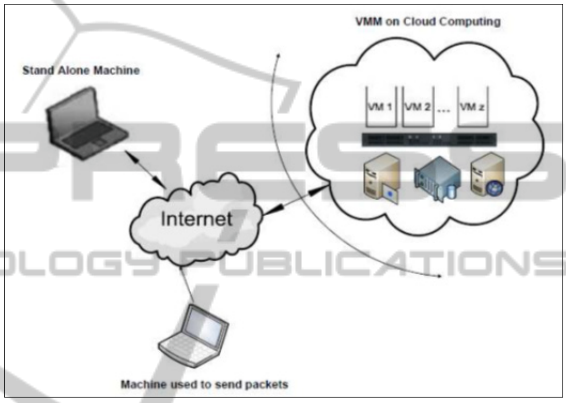

Figure 2: Experimental Environment Set up.

A custom script was executed in the test machine to send packets to target servers requesting for IP timestamps replies. This request was repeated until 5,000,000 times. IP packets data structure with timestamps request options are as per Figure 3 and IP timestamps option packets structure that were cosntructed is as per Figure 4. These requests were sent to the target hosts.

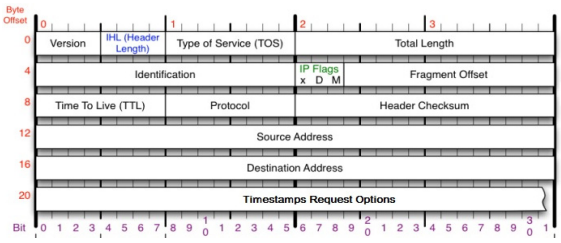

Figure 3: IP Packets Data Structure.

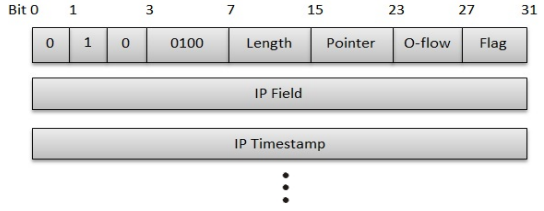

Figure 4: IP Timestamps Option Packets Structure.

The specifications for the experiment environment are shown in Table 1, Table 2 and Table 3.

Table 1 Experiment Environment for Real OS.

OS	Linux Ubuntu 12.04
CPU	Linux Ubuntu 12.04
Memory	1GB
Hardware	Dell Power Edge

Table 2: Experiment Environment for VMWare.

OS	Linux Ubuntu 12.04
VMM	VMWare vSphere Hypervisor (ESXi) 5.1.0
CPU	Intel Xeon E5-2440
Hardware	Dell Power Edge
Memory	(Virtual Allocation) 1GB
Storage	(Virtual IDE) HDD 16GB

Table 3: Experiment Environment for VirtualBox.

OS	Linux Ubuntu 12.04
VMM	Oracle VirtualBox 4.3.12
CPU	Intel Xeon E5-2440
Hardware	Dell Power Edge
Memory	(Virtual Allocation) 1GB
Storage	(Virtual IDE) HDD 16GB

For the target server, VM software from VMWare, VM vSphere 5 was emulated in the VM environment as full virtualization VM system and Linux Ubuntu 12.04 was emulated as the OS on the VM. 5,000,000 times of request packets were sent to the target server to obtain its timestamps in the IP packets reply. Later the same experiment was also done using Oracle Virtual Box with Linux Ubuntu 12.04 as the emulated OS. Multiple IP timestamps requests were sent constantly to remote hosts as attempts to measure the differences in timestamps reply characteristic received from real machine and VM. The differences were measured to the nearest microsecond. We chose high performance Intel(R) Xeon(R) Processor E5-2440 with 6 cores and 2.40GHz clock speed as the host machine for the VM in our test environment. Timestamps replies that were received at the request machine were compiled to the nearest microsecond as data in a CSV file. In the case of VM, since there will be VM interface between the CPU and the network interface, it is expected that there will be a small delays for issuing the timestamps between the requests. On the other hand, it will not be an issue in real machine since it will only have CPU and network interface interactions. Thus, as the results from analysing and comparing the timestamp replies data between VM and real machine, it is expected that the timestamps value from VM will change more frequently compared to the timestamp replies in real machine.

5 EXPERIMENTAL RESULTS AND ANALYSIS

In our previous study we had proven that para-virtualization VMs are clearly detectable remotely by analysing the replies from IP timestamps. The conclusion was done based on the collected timestamps data in IP and ICMP replies by looking at timestamps stamping differences. In this study, we performed testing using high performance machine by implementing full virtualization technique using VMWare ESX and VirtualBox. The timestamps reply consists of 32 bits but only could be display in 8 digits of timestamps in hexadecimal and 10 digits of timestamp in decimal number. The collected IP timestamps data was analysed to determine its characteristic behaviour. In the data analysis, we define N as count for how many times same timestamps was replied from the targeted server. For real machine, we observed that 62% of the timestamps replies were 5 times of the same timestamps value and 25% of the timestamps replies were 4 times of the same timestamps value. However, we also observed that for VM environment the behaviour of timestamps stamping are different which are, lesser same timestamps were sent as replies when the timestamps reply request were sent to the requestor. We could clearly see the behaviour pattern differences in this experiment as shown in Table 4. Figure 5 has shown the comparison of real environment and VM environment, for how many time the same timestamps reply were sent. In real machine, more than 60% of timestamps are stamped for 5 times. In contrast, we could see that there were no 5 times same timestamps reply in VirtualBox and VMWare ESX.

Table 4: Experiment Results for Full Virtualization.

Count N	Real Machine (%)	VirtualBox (%)	VMWare (%)
7	2	0	0
6	3	0	0
5	62	0	0
4	25	1	36
3	5	19	55
2	2	47	8
1	0	33	2

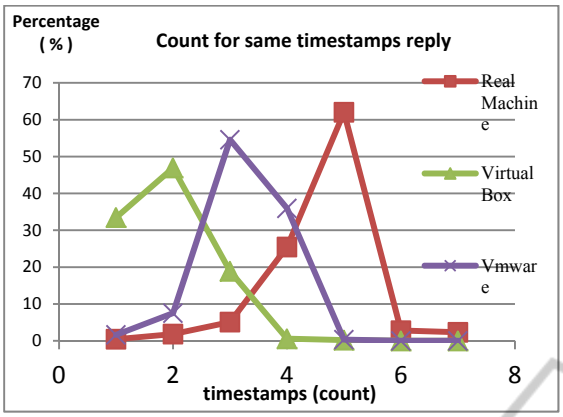

Figure 5: Count for same timestamps reply.

The reason is because VM sometime interrupted timestamps operations to complete other operations and make the time taken to complete job longer than real machine. Even though in full virtualization that simplifies migration and portability, the remote detection by using IP timestamps packet reply still could be observed and this could reveal the environment that one system is running on. Malware will be able to detect the environment that they are running on and could choose not to run at all or manipulate the running environments for the attack. Full virtualization offers the best isolation and security for virtual machines, however, remote detection by using network timestamps need to be addressed to resolve the VM vulnerability. Since that machine that was used to install the VM is continuously improved, the difference according to performance might be different. More tests need to be done constantly and periodically to verify the current situation and detect vulnerability.

6 CONCLUSIONS

Building a transparent VM is still a difficult task, as shown from the results of this research where remote detection method was discovered in full virtualize-tion technique. How VM could be detected has significant value that requires extensive studies and research in order to prevent any security loop holes that could exploit the vulnerability of VM including security holes that caused by any possible detection method. By analyzing all possible detection methods, which will be the ideal expectation, countermeasures could be proposed and implemented for creating better secured VM in cloud computing environment. Our study explores detection methods by looking it from perspective of detecting VM existence by performing timestamps analysis in full virtualization using IP timestamp option, so that vulnerability that caused by ability to detect VM existence from timestamps data could be addressed. As a future work, we would like to perform more tests using various machines and also in grid and cloud test bed to get more data characteristics for comprehensive analysis. This research shows that behavior between IP timestamps packets reply from the most popular technique of choice, VMWare ESX and VirtualBox could be differentiated remotely and this could be potentially used as a VM detection method before attack vector chosen to compromised the target host. Based on our discussion in previous section that timestamps reply will be different in VM and stand-alone machine, as a future work, we would like to propose a mechanism to change timestamps reply in stand-alone machine to look similar with VM timestamp reply. This mechanism should be able to hide the differences between timestamp reply in VM and non-VM and prevent it from being used as method to detect either that the machine is running on VM environment or not. Later on, based from this research, we will also perform more studies about malware behavior and its relationship with VM discrepancies in both full virtualizations and para-virtualization. We are aiming to set up various scenarios for both full and para-virtualization and obtain various characteristic of data. The results will contribute on providing more secure cloud computing services. The approach that we proposed will contribute to VM environment prevention of detection based on remote timestamps reply. Our approach will decrease the possible detection method that could lead to manipulation for cloud computing environment and the possible attack vector. Base on the finding, we would like to propose the solution by applying it to cloud computing security policy framework for cloud computing.

REFERENCES

Anthes, G., "Security in the cloud." Communications of the ACM 53(11): 16-18. (2010).

B. Lau and V. Svajcer. “Measuring virtual machine detection in malware using DSD tracer”. Journal in Computer Virology, 6(3), 2010.

Bernd Grobauer, Tobias Walloschek, and Elmar Stocker. 2011. Understanding Cloud Computing Vulnerabili-

ties. IEEE Security and Privacy 9, 2 (March 2011), 50-57.

J. Crandall, G. Wassermann, D. Oliveira, Z. Su, S. Wu, and T. Chong. “Temporal search: detecting hidden malware timebombs with virtual machines”. In ASPLOS-XII: Proceedings of the 12th international conference on Architectural support for programming languages and operating systems, pages 25–36, New York, NY, USA, 2006. ACM Press.

J. Franklin, M. Luk, J. M. McCune, A. Seshadri, A. Perrig and L. van Doorn. “Remote detection of virtual machine monitors with fuzzy benchmarking”. SIGOPS Oper. Syst. Rev., 42(3):83–92, 2008.

J. Watson, “Virtualbox: bits and bytes masquerading as machines,” Linux Journal, vol. 2008, no. 166, p. 1, 2008.

K. Miyamoto, H.Tanaka, “Proposal of Effective Detection Method of VMM without Feature Database”, Information Processing Society of Japan, Vol. 52. pp. 2602-2612, 2011. (Japanese).

Karen A. Scarfone and Peter M. Mell, Guide to Intrusion Detection and prevention Systems (Idps), technical Report NIST Gaithersburg, MD, United States, 2007.

Matrazali Noorafiza, Hiroshi Maeda, Toshiyuki Kinoshita, Ryuya Uda: Virtual machine remote detection method using network timestamps in cloud computing. ICITST 2013: 375-380.

Mills D. Network Time Protocol (version 3): specification, implementation and analysis. Technical Report RFC 1305,Network Working Group; March 1992.

Nance, Kara, Hay, Brian, Bishop, Matt "virtual machine introspection." IEEE Computer Society. (2008).

P. Ferrie, “Attacks on More Virtual Machine Emulators”, Symantec Advanced Threat Research, 2006.

P. Padala, X. Zhu, Z. Wanf, S. Singhal, and K. Shin, "Performance evaluation of virtualization technologies for server consolidation, HP Labs, Tech. Rep. HPL-2007-59, 2007.

R. Buyya, C. Yeo, S. Venuopal, J. Broberg, and I.Brandic, “Cloud Computing and emerging IT platforms: vision, hype and reality for delivering computing as the 5th utility”, Future Generation Computer Systems, pp. 599-616, 2009.

T. Garfinkel, K. Adams, A.Warfield, J. Franklin, “Compatibility is Not Transparency: VMM Detection Myths and Realities”, Proceedings of the 11th Workshop on Hot Topics in Operating Systems (Hot OS-XI), 2007.

T. Kohno, A. Broido and K.C. Claffy. “Remote physical device fingerprinting”. In SP ’05: Proceedings of the 2005 IEEE Sympossium on Security and privacy, pages 211-255, Washington, DV, USA, 2005.

The Internet Engineering Task Force Darpa Internet Program Protocol Specification, http://www.ietf.org/rfc/ (Access date: 23 September 2014).

U. Bayer, C. Kruegel, and E. Kirda. “TTAnalyze: A Tool for Analyzing Malware”. In 15th Annual Conference of the European Institute for Computer Antivirus Research EICAR), 2006.

VMWare, “Understanding Full virtualization, Para-virtualization and hardware Assist,” 2007. [Online]. Available: http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf (Access date: 23 September 2014).

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS