Cloud security problems caused by virtualization technology vulnerabilities and their prevention

Article · January 2014

CITATIONS
4

READS
2,094

2 authors:

Rudolf Hoermanseder
Johannes Kepler University Linz
13 PUBLICATIONS **24** CITATIONS
SEE PROFILE

Markus Jäger
Pro2Future GmbH
36 PUBLICATIONS **121** CITATIONS
SEE PROFILE

CLOUD SECURITY PROBLEMS CAUSED BY VIRTUALIZATION TECHNOLOGY VULNERABILITIES AND THEIR PREVENTION

Rudolf Hörmanseder, Markus Jäger

Institute for Information Processing and Microprocessor Technology (FIM)
Johannes Kepler University Linz (JKU)
{rudolf.hoermanseder, markus.jaeger}@jku.at

Keywords

Cloud Computing Security, Virtual Machine Based Rootkits, Virtualization Security, Virtual Machine Introspection, Self Cleansing for Intrusion Tolerance

Abstract

In public "Infrastructure as a Service" (IaaS) clouds, several Virtual Machines (VMs), owned by different customers, are executed on a single physical machine. This multi-tenancy gives malware the chance to not only take control over a single VM; but additionally the malware can start attacks against other VMs on the local hardware and also against the Hypervisor. If the worst case occurs and an attack on the Hypervisor succeeds, all the VMs on the physical machine are compromised.

We present risks and known attacks on virtualization infrastructures and discuss approaches for prevention, specifically Virtual Machine Introspection (VMI) and Self Cleansing for Intrusion Tolerance (SCIT).

1. Introduction

Virtualization Security in a public cloud providing IaaS [26] is very important, because of e.g. Multi-Tenancy - executing multiple VMs on one physical machine offers many possibilities to malware for attacks. This paper is designed as a motivation and a starting point for readers to think about the security pros and cons of virtualization.

Using the services of a Cloud Provider (CP), you as a customer cannot identify on which physical server your VM is located and it is deployed together with instances of VMs of other customers. Even using many instances of VMs, you never know, whether they (or some of them) are located on the same physical machine or not. This is suboptimal concerning availability (e.g. distribution) and security (no "foreign" VMs on the same hardware). Additionally you have (almost) no chance to secure your VM against the virtualization layer and the Virtual Machine Monitor (VMM).

The Federal Office for Information Security in Germany wrote in [3]: *"…With a few exceptions, no attacks on the hypervisor have yet appeared in the wild [...] - they have only been described in theoretical terms or as proof-of-concept. Should an attack succeed, however, the consequences are devastating."*

"Shared Technology Vulnerabilities" are listed only at the last (9^{th}) position in the current list of "Cloud Computing Top Threats" of the Cloud Security Alliance [6]. This threat, which also

includes virtualization vulnerabilities, occupied position 4 three years ago. This is good news. However, the text describing this type of vulnerability also includes sentences like *"This vulnerability is dangerous because it potentially can affect an entire cloud at once".*

The conclusions from these assessments are somewhat unpleasant: the problem is tremendous because of the multiplier effect of the many affected users. Due to the increasing use of cloud computing and based on a concentration of few large providers, one can fear/assume that the impact of a (single) security incident will affect more and more users simultaneously. So far, the concerted joint efforts of manufacturers, CPs and researchers have prevented successful against (known & public) attacks. Thus it proves exceedingly difficult to successfully attack a VMM, nevertheless the possible impact – if it really happens – is huge.

Our aim is to illustrate security risks concerning the physical implementation through the assumption that one instance of an infected VM gives malware the possibility to break out and infect other instance of Virtual Machines or in the worst case, the VMM itself. We concentrate on data-leakage attacks, which often stay unrecognized for a long time, as they typically do not prevent normal operation. Because of the high speed of machine-internal communication these attacks may also be very fast. This e.g. shares some similarities with the Heartbleed attack [15, 10] (CVE-2014-0160).

2. Security Risks by Virtualization

In this section we discuss a scenario showing several risks concerning cloud architectures, known security vulnerabilities implied by the architecture of different virtualization-types, and the danger posed by VMBRs (Virtual Machine Based Rootkits).

The architecture of many commercial cloud providers is not public. So we consider a simplified cluster as an example.

Below we describe several individual risks, enumerated from a to f as shown in Figure 1:

a.) VM to VM over the Internet: Once infiltrated, malware can use the power of a VM to operate against the Internet and other targets, but it can also execute attacks against VMs on the same physical system.
b.) From the Internet: If the attacker does not have access to a VM, he has to come from the outside. Here it does not matter if the origin is really from the outside or from another VM.
c.) VM to the outside: This type is concerned by attacks like (D)DoS to other machines or by misuse as part of a botnet (sending Spam-Mails, etc.).

These three types (a-c) are classical attacks; we do not describe them in detail.

d.) VM to VM directly, on one physical machine: Though this type of outbreak is a possibility, we do not know of any incidents, where one VM has infected another VM without any communication over the Internet or going through the VMM.
e.) VM to external storages: If a VM is infected and has access to a cluster-storage to which other VMs also have access, this is one way for supposedly safe machines to get infected. For example you also can imagine a type of shared-folder option like in common VM solutions. Another problem concerning this attack is the storage of VMs (while turned off) in an external site-storage. While the machine is turned off, there's no actual problem (and no active defence-mechanisms), but there still exists the risk of restoring an infected VM back on a formerly clean VMM.
f.) VM to VM over the VMM: This is probably the most dangerous outbreak of malware used in a VM. If malware is able to gain control over the VMM via "Virtual Machine Escape" (or

via a VMBR undermining the VMM), the security of the physical machine is gone and therefore the safety of this part of the cloud is not guaranteed any more.

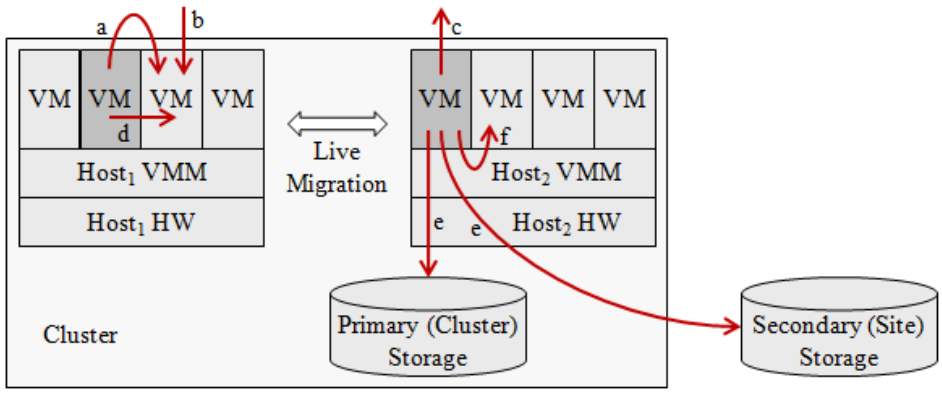

Figure 1: Simplified Cluster as Cloud Infrastructure; several terms follow Apache CloudStack [8]

Security implications from weak implementations of core virtualization requirements are discussed in [32]. It describes attack types such as VM-Escape (in general) VM-Escape to host (f) and VM-Escape to VM (d).

In 2006, a research group from Microsoft Research and the University of Michigan demonstrated the first instance of a Virtual Machine Based Rootkit (VMBR). This new type of Rootkits has the goal to undermine the active Operating System (OS) and hoist it into a VM without getting the user's attention. The paper "SubVirt" [21] describes step-by-step how malware is able to do this.

After this innovative functionality of malware, Joanna Rutkowska from Invisible Things Lab presented another VMBR which is able to hoist the running OS into a VM on-the-fly - with no need to reboot. This VMBR has the name "Blue Pill" [34]. The Rootkit allocates so much memory that sensitive kernel-code must be switched to the hard drive, where it can be modified by the Rootkit. This modification is not possible while the code is in the main memory. When the code is reloaded into the main memory and executed, the Rootkit gains control over the system.

Two years later, at Black Hat 2008, the "Xen 0wning Trilogy" was presented. In a three-step-presentation and live demo it was shown how to subvert the Xen Hypervisor [41] using DMA (direct memory access) attacks and the "Xen Loadable Module Framework". They also gave a guidance of how to prevent a Xen Hypervisor of being subverted. There is also a comparison between Xen, Microsoft's Hyper-V and VMware's ESX/vSphere. Known vulnerabilities concerning these security breaches are listed as CVE-2007-4993 and CVE-2007-5497 [10]. Other attacks against Intel and Xen are provided in [42].

Further risks concerning our scenario are the so called "Virtual Machine Escapes" - the threat of malware being able to break out of its isolated environment, the VM, and taking control over its VMM. One exploit of this type affecting VMware is listed as CVE-2008-0923 [10]. A working exploit was "Cloudburst", where the guest systems were able to write into the Hypervisors file system through the "Shared Folder" functionality.

Newer exploits of VM-Escapes are from 2012: the VUPEN Method [39] and other incidents, e.g. where a manipulated VMDK (Virtual Machine Disk) descriptor file was uploaded to the VMM and was able to load VMware ESXi system files directly to the VM [13]. An incident on Xen's 64-bit para-virtualized Hypervisor is shown in [36].

Closing this section we want to draw some attention to a few related publications on how to prevent such attacks: [35] treats many general securing aspects. Other authors [4, 14, 20] describe the usage

of Virtual Intrusion Detection/Prevention Systems (vIDS/vIPS) or deploy security software in a special privileged VM (SecVM) and the approach of VMI - observing VMs from the outside.

3. Possibilities to protect and avoid

An attack against the VMM can originate from many sources, for example if there is an intrusion into the central cloud management or if the customer's accounts are hacked. Because this paper focuses on how to prevent breaking into the VMM from a customer VM, we concentrate on the VMs and the VMM here.

Protection measures against "VM-Escape to host" (see above) are addressed in [32], who proposes VMM integrity checking, attestation and security-aware design, programming and testing for (at least) EAL5 level according to Common Criteria [9] as security measures. This includes (amongst others) VMM patching, host security measures (similar to physical machines) and *"...measures to detect the malicious code in the VM (as legitimate software does not need such functionality)."* Therefore the following chapter "VM Introspection" deals with detection of this malicious code in the VM.

The following subsections discuss two examples of security concepts. These two concepts do not provide a substitute for other precautions, but must always be seen as an additional step to reach a higher level of security. Additionally, note that they do not cover all attack vectors described above, but concentrate on selected aspects only.

3.1 Can Virtual Machine Introspection (VMI) help against VMM flaws?

We have now talked about several security challenges in the cloud. There are also security benefits because of virtualization. The following example concerning VMI shows one of these technical benefits. [14] states that "*Being able to directly inspect the virtual machine makes it particularly difficult to evade a VMI IDS since there is no state in the monitored system that the IDS cannot see."* And [22] proposes *"... that IDPS should be placed at the hypervisor level, as that choice provides many security benefits compared to other designs."* (IDPS = Intrusion Detection and Prevention System).

If an attacker takes over a VM of a legitimate user to misuse it as the origin of a "new" attack against the VMM, a VMI-based IDPS has a good chance to detect this intrusion into the VM. This positive statement results from the assumption that the attacker will use malicious code to infiltrate the VM and that this malware is already known, so that the VMI-IPS can deal with it. An example can be found in [4]. Their Rootkit detection monitors kernel pointers, which are typically modified by Rootkits. But even in this case, the authors consider reducing the number of false positives by switching from whitelisting to blacklisting. Thus in many cases only already well-known Rootkits will result in an alarm.

The occurrence of a detection method leads straight to the next problem: how can/should the IDS, and thus the CP, react to the suspected threat? As an example, [29] shows different VMI approaches and analyzes technical concepts in the context of their VIX-tool. Nevertheless, legal aspects *"... where the monitoring process results or effects can have real and serious legal consequences"* are listed in "Future investigations". Especially if the alarm was a false-positive, an immediate deactivation of the VM may lead to claims for compensation.

There are additional legal issues: to what extent is the CP allowed to analyze the main memory, storage, network traffic and all other activities of a customer's VM, even if the CP does not need

this information for billing? Does a customer really desire that the CP monitors all activities *inside* the VM, including all the files and data written to ephemeral and persistent storage in addition to all network activities (which are typically inspected by a firewall at least)? Of course, several customers will accept an offer from their CP for "Security as a Service". Nevertheless, there will be other customers as well.

It is evident that the cloud customer also has duties, such as to implement appropriate safety measures, to use security tools and to keep the software up-to-date. However, these obligations exist regardless of whether or not the system is operated in a cloud or directly on hardware located at the premises of the company. In the chapter *"Cloud computing providers and you: a shared responsibility"* [19] states for instance that *"By design, the IBM staff and tooling do not access or scan a client's virtual environments."* Additionally the responsibility for all patch management lies in the hands of the client. In contrast to this, e.g. if the CP just automatically installs patches on a virtual image at rest instead of protecting VMs until they can be updated [7], how can customers use software which is only guaranteed to work for certain patch-levels? So in general security responsibility must remain with the customer.

Securing the VMs against attacks from the outside is an important step to protect a cloud. VMI gives the CP an additional line of defence, which is located between network security and the VMM itself, by ensuring that "unsecured" VMs are also observed. Because of customer privacy and legal issues, in general only attacks from a whitelist (also known as *knowledge-based* or *signature-based* detection) will result in an immediate "hard" reaction comparable to an IDPS. If an attack is already known, fully analyzed, and included in the signature-database of the IDPS, the probability of false alarms will be typically low enough to take immediate steps. A behaviour-based IDS can find attacks which are unknown so far, *"... but its accuracy is a difficult issue."* [12]. This is still (to some extent) true for cloud environments: the deviation from a "normal" behaviour is often difficult to measure, because the normal behaviour of customer's VMs may change rapidly without any further notice and varies between customers.

Additionally, the detection of Rootkits and backdoors in VMs do not help against malicious customers. IaaS customers always have the permission to access and modify their environment inside the VM, e.g. to install new software like a backdoor.

CPs apply updates to their core infrastructure on a regular basis. Therefore, we feel confident that known vulnerabilities inside the VMM, for which IDPS signatures are available, have been patched already. Anyhow, IDPS signatures, which recognize attacks against the VMM, are by no means redundant, as they offer the chance to identify malicious customers or VMs which have already been subverted by attackers.

Due to the rapid patch update cycles, it can be stated: a dangerous attack must be so new that there exist no IDPS rules and no patches against it. Thus it must be (at least) a zero day attack. Therefore – as already noted in the introduction – it is exceedingly difficult to successfully attack a VMM. However, if these circumstances occur, the VMI cannot rely on IDPS signatures to protect the VMM, even if new attacks or variants are sometimes detected by them [16]. Additionally, the logs of behaviour-based IDPS need to be analyzed carefully to detect hints of any such incident, an increased workload requiring specialised personnel.

It should also be noted that VMI, as any security measure in general, is not for free. Depending on the depth and scope of operation, a VMI-based IDPS may require a lot of performance: how will the CP charge these costs? Despite the clear separation of functionality, e.g. into privileged VMs, in-depth VMI makes the VMM more complex and thus eventually also more error-prone.

The short summary is as follows: Alarms of an IDPS, indicating an attack of a VM to the VMM, can only lead to direct switching off (or freezing) the attacking VM when the probability of a false positive is very low. Unfortunately, for new zero-day attacks this is rarely the case. So the benefit, that it is hard(er) to circumvent a VMI-based IDPS than a standard host-based system, is often not as large as expected. Thus customer's VMs in public IaaS clouds must still be considered hostile!

3.2. Can SCIT reduce the effects of VMM flaws?

SCIT [37] (Self Cleansing for Intrusion Tolerance) restarts a system from a known good state in certain intervals for security reasons. This concept acts on the pessimistic assumption the worst case happens and that an incident may occur fully undetected, and tries to remedy it through a "reset". An overview on other intrusion-tolerant system architectures can be found in [30].

SCIT shares many similarities with "Software Rejuvenation" and "Recovery Oriented Computing". These concepts typically differ in the targets and often concentrate on dealing with software-, hardware- or human operating-errors [18].

Assumptions: SCIT follows a pessimistic point of view and a low degree of trust: if a computer system has been exposed to potentially successful attacks, it is classified as penetrated and compromised. This assumption is maintained even when there are no signs of actual attacks or intrusions. Thus the considerations are conceptually independent from other precautions like IDPS, or special security layers and trusted hardware, which try to ensure the integrity of a system. If a server offers services to a public network, it is always exposed to attacks. We assume the same situation for VMMs running VMs in a public IaaS cloud on them.

SCIT assumes that an intruder needs time to infiltrate the system and to do real damage; it reduces the intruder residence time by decreasing the server exposure time, the time a server is online until the next new (re-)installation. [18] shows in an example that this period can be reduced to less than 5 minutes using the same resources as with typical primary-backup-redundancy.

Cleansing: As in everyday life, a suspicious server is taken out of service and is (re-)installed to a clean state. SCIT performs automatic reinstallations at periodic intervals. While e.g. [1] suggest special hardware in this case, a typical solution in the context of CPs will rely on secure remote boot from a network interface which is connected to a dedicated out-of-band management network. Thus the cleansing operation may consist of a simple image-copy during reboot or just use a reboot to clear information stored in an overlay file system in RAM. Additional security related tasks, e.g. integrity checks, might be part of the cleansing (offline) phase. At this time for instance a simple file-content comparison could potentially reveal successful resident intrusions, even if they managed to circumvent the built-in integrity verifier mechanisms of the VMM. E.g. the Intel Trusted Execution Technology (TXT) supports and promotes such integrity verification of the image during boot [17].

Availability: While one server is offline, (an)other server(s) must take over its services (=VMs) to preserve availability. Many papers about SCIT describe that it is therefore suited best for stateless systems and supports stateful services only for short-termed sessions. So SCIT supports services with no or almost no state like stateless packet filtering, static HTML webservers [2] or DNS servers. Additional papers concerning this topic can be found e.g. in [1, 18].

SCIT for VMMs: In contrast, the cost for taking a physical server in the cluster down will be remarkable, even if the VMM is assumed to be stateless to a large extent: similar to [38] the VMM should write logs and performance data to a central storage system. Nevertheless, in most cases several active VMs must be migrated to other servers by life-migration [5]. The migration of all active VMs from a server is similar to the case that health monitoring based on hardware sensors

predicts a hardware problem [27], but occurs far more frequently and not only in exceptional circumstances. Automatic hot stand-by reduces this time on the expense of efficiency.

Policy based SCIT: Because of the expected large effort for VM migrations, a policy based solution – as opposed to simply taking servers down for cleansing in fixed (or better random) intervals – is recommendable. [43] adapts SCIT to withstand DoS attacks based on historical data. [25] uses an adaptive SCIT policy based on response delays in combination with CVE scores and file integrity for attack prediction and the decision process, how many and which system(s) should be online. Because we do not want to go any deeper into resource scheduling strategies and concentrate on stealth attacks, we use the description in [24]. To secure the VMM against yet unknown attacks, our scoring policy should not solely rely on data from CVE and CVSS (Common Vulnerability Scoring System [11]). And if the integrity-verifier, which is built into the VMM or the cleansing process, recognizes that e.g. the /etc/passwd file has been changed, this will result in a "red alert" and an immediate response, and not just reduce this server's online time.

SCIT policy based on VMs trustworthiness: We propose to estimate the threat of a VM on the VMM based on the runtime of the VM (as in "simple" SCIT) and on customer-specific criteria of the VM owner. There are several indications for a CP to classify customers. Technical aspects depend on the policy of the provider and how much technical information about the VMs is (allowed to be) available by the customers.

Examples for customer classification: history (trouble-free customer for a long time), line of business (e.g. security company is rated higher), credit reports about the customer, attack history … Examples for technical classification: number of CPUs, size of memory, open TCP/UDP ports, OS, ... including all data which is used by CP for billing.

On the one hand the rating is inherited from the customer to his/her VMs. This aspect is especially important if the provider does not collect any security-relevant technical information from the VMs. On the other hand, technical information about VMs influences the rating of the customer at least in the long run: what should the CP think of a customer whose VMs repeatedly create assured security alarms? This again has strong similarities with [24, 25, 31]. The simplest relation between the remaining online time of a VMM until the next cleansing and the guest VMs can be described as follows: in periodic (real) time intervals, the remaining online time is decremented by the sum of all the ratings of all VMs which are active on this server (higher rating means lower trustworthiness here).

SCIT for VMMs and green computing: If several servers in a cluster are automatically turned off during under-utilization to save power [33], this is an ideal environment for SCIT. The cleansing process can be seen as an extension to power-aware scheduling. Nevertheless the algorithms have to be modified, especially because in SCIT every server has to be taken offline regularly; it is not enough to re-initialize an arbitrary server, as for power-saving reasons.

Heterogeneity / Diversity: When an attacker is locked out because of cleansing, it is very likely that he will try to exploit the same vulnerability again. Attackers will learn to execute their offenses more quickly e.g. by scripts. SCIT may use diversity in software or configuration to additionally discourage the intruder by the fact that (hopefully) only a few systems are susceptible to a dedicated vulnerability and that the majority will not show the same flaw - and the attacker cannot predict where he will end up after a cleansing.

Using different virtualization platforms within one cluster adds problems: problems that can be anticipated with the compatibility of the images and in particular the necessary live-migration between different vendors make this all but impossible. A promising general approach is "Security through Diversity", by which – instead of the distribution of identical images – every instance is

“custom built” (=compiled with randomization of addresses, instructions etc.) and thus unique. [23, 40] provides a good overview of this concept.

“Rating”: SCIT without policies and without diversity can be seen as a kind of “brute-force” mitigation measure, if the analysis within the cleansing process does not result in security-relevant information. Fast individual attacks, as Heartbleed, still are possible even if SCIT includes diversity. Nevertheless, many attacks will need longer to succeed if several variants are not prone to the same flaw. Additionally there is the hope that an attacker (because of SCIT!) "by mistake" also attacks non-vulnerable systems or has to resort to more suspicious behaviour and the reactions of these systems give evidence of the ongoing attack. E.g. a stealth attack on one vulnerable VMM may result in IDS warnings or a denial-of-service (bad enough!) on another VMM.

3.3 Combination of IDPS & SCIT

[28] develops a cost model for IDPS false positives/negatives and combines this with SCIT. The results show that the costs of a false negative are reduced by SCIT; thus the number of false negatives may be increased, rendering false positives more unlikely. We already mentioned that if we want to turn off an attacking VM, the probability of a false positive must be very low. For this reason SCIT can support and enhance the IDPS in our scenario.

4. Conclusion and further Work

The classifications in the text, how long a VMM should be online before cleansing, just show the basic approach. We did not simulate advanced (adaptive) concepts until now; this is ongoing/future work. These calculations also have to integrate the security risk, if an attacking VM is migrated from one physical computer to another, because this may result in a new field of activity for the attacker. Considerations to isolate potentially attacking VMs, if necessary by "putting all the presumably rotten eggs in one basket” have not been investigated yet too.

The paper lists several examples of known attacks against the virtualization layer. And it would be too optimistic that in the future there will be less or even no errors in this area.

The discussion of the selected two security measures shows that both increase security, but it also illustrates their limits. The comments on VMI point out that reliable recognition of zero-day attacks in IPS is still an ongoing major concern. This not only involves topics such as artificial intelligence or big data, but legal issues as well: threat detection services relying on telemetry data of multiple companies must not compromise their privacy.

The combination of SCIT and diversity is particularly promising. Concentration trends in hardware and software cut costs because the development expenses can be distributed to many customers. But this trend simultaneously promotes a kind of monoculture that renders the use of heterogeneous products more difficult and expensive. Thus our hope is directed towards widespread and extensive use of machine-made automatic diversity. As might be expected, the combination of several measures (in our example IDPS + SCIT + diversity) gives the best overall result. Thus only permanent and continuing efforts and activities of all parties concerned will ensure that the status quo of security is maintained. From a technical view this includes (without any claim to completeness) research and development in hardware security mechanisms, tamper-proof and encryption devices, OS, network, and application security, computer languages and compilers and runtime environments, software testing, algorithms for AI, software design and engineering processes, … Finally we all are called to arms to maintain security!

5. References

[1] Arsenault, D., Sood, A., Huang, Y., (2007), "Secure, Resilient Computing Clusters: Self-Cleansing Intrusion Tolerance with Hardware Enforced Security (SCIT/HES)", ARES 2007 2nd International Conference on Availability, Reliability and Security, pp. 343-350

[2] Bangalore, A.K., Sood, A.K., (2009), "Securing Web Servers Using Self Cleansing Intrusion Tolerance (SCIT)", DEPEND Second International Conference on Dependability, pp. 60-65, IEEE 2009

[3] BSI (2011): „Security Recommendations for Cloud Computing Providers" Article Number: BSI-Bro11/314e https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Minimum_information/SecurityRecommendationsCloudComputingProviders.pdf?__blob=publicationFile, p. 25

[4] Christodorescu, M., Sailer, R., Schales, L., Sgandurra, D., Zamboni, D. (2009), "Cloud Security Is Not (Just) Virtualization Security, a short paper", CCSW 09 Proceedings of the 2009 ACM workshop on Cloud computing security; pp. 97-102, 2009

[5] Clark, C., et.al., (2005), "Live migration of virtual machines", Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation NSDI'05, vol. 2, pp. 273-286, USENIX Association Berkeley

[6] Cloud Security Alliance (2013-1): "The Notorious Nine: Cloud Computing Top Threats in 2013" http://www.Cloudsecurityalliance.org/topthreats/

[7] Cloud Security Alliance (2011), "Security Guidance for Critical Areas of Focus in Cloud Computing, Version 3.0", 2011 https://Cloudsecurityalliance.org/research/security-guidance/#_v3

[8] CloudStack Administrator's Guide (2014), https://Cloudstack.apache.org/docs/en-US/Apache_CloudStack/4.2.0/html-single/Admin_Guide/index.html#Cloud-infrastructure-concepts

[9] Common Criteria (2014), "Common Criteria: New CC Portal", http://www.commoncriteriaportal.org/

[10] Common Vulnerabilities and Exposures, http://web.nvd.nist.gov

[11] NVD Common Vulnerability Scoring System Support v2, http://nvd.nist.gov/cvss.cfm

[12] Debar, H., Dacier M., Wespi, A., (2000), "A Revised Taxonomy for Intrusion Detection Systems", Annals of Telecommunications, 55 No. 7-8, pp. 361-378, 2000.

[13] Finn, A., (2012), "Serious Break-Out Security Flaw Found in VMware Cloud", http://www.aidanfinn.com/?p=13363

[14] Garfinkel, T., Rosenblum, M., (2003), "A virtual machine introspection based architecture for intrusion detection", Proceedings of the 2003 Network and Distributed System Security Symposium

[15] Heartbleed (2014), "OpenSSL Vulnerability, US-CERT" http://www.heartbleed.com http://www.us-cert.gov/security-publications/Heartbleed-OpenSSL-Vulnerability

[16] Holm, H., (2014), "Signature Based Intrusion Detection for Zero-Day Attacks: (Not) A Closed Chapter?", HICSS 47th Hawaii International Conference on System Sciences, pp. 4895-4904

[17] Intel (2014), "Intel® Trusted Execution Technology (Intel® TXT) Enabling Guide", Version 1, 330139-001US, https://software.intel.com/en-us/articles/intel-trusted-execution-technology-intel-txt-enabling-guide

[18] Huang, Y., Arsenault, D., Sood A., (2006), "Closing Cluster Attack Windows through Server Redundancy and Rotations", Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid CCGRID'06, Vol. 2, IEEE

[19] IBM Global Technology Services (2011), "Security and high availability in cloud computing environments", Technical White Paper MSW03010-USEN-00, June 2011 http://www935.ibm.com/services/za/gts/Cloud/Security_and_high_availability_in_Cloud_computing_environments.pdf

[20] Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J., (2010), "Emerging Security Challenges of Cloud Virtual Infrastructure", Proceedings of APSEC 2010 Cloud Workshop, Sydney, Australia, 30th Nov 2010

[21] King, S.T., Chen, P.M., (2006), "SubVirt: implementing malware with virtual machines", Proceedings of the 2006 IEEE Symposium on Security and Privacy

[22] Laniepce, S., et.al., (2013), "Engineering Intrusion Prevention Services for IaaS Clouds: The Way of the Hypervisor", 2013 IEEE 7th International Symposium on Service-Oriented System Engineering, pp. 25-36

[23] Larsen P., Brunthaler S., Franz M. (2014): "Security through Diversity: Are We There Yet?", IEEE Security & Privacy, Vol. 12 No. 2, pp. 28-35

[24] Lim, J., Doo, S., Yoon, H., (2013), "The Design of a Policy-Based Attack-Resilient Intrusion Tolerant System", in Information Technology Convergence: Security, Robotics, Automations and Communication, part 2, pp. 435-443, Springer 2013

[25] Lim, J., Doo, S., Yoon, H., (2013), "The Design of a Robust Intrusion Tolerance System through Advanced Adaptive Cluster Transformation and Vulnerability-Based VM Selection", MILCOM Military Communications Conference, pp. 1422-1428, IEEE 2013

[26] Mell, P., Grance, T. (2011), "NIST Special Publication 800-145: The NIST Definition of Cloud Computing", Recommendations of the National Institute of Standards and Technology

[27] Nagarajan, A.B., et.al., (2007), "Proactive fault tolerance for HPC with Xen virtualization", ICS '07 Proceedings of the 21st annual international conference on Supercomputing, pp. 23-32, ACM 2007

[28] Nagarajan, A.; Quyen Nguyen; Banks, R.; Sood, A., (2011) "Combining intrusion detection and recovery for enhancing system dependability", 41st International Conference on Dependable Systems and Networks Workshops (DSN-W), pp. 25-30, IEEE/IFIP

[29] Nance, K., Bishop, M., Hay, B., (2008), "Virtual Machine Introspection: Observation or Interference?", IEEE Security & Privacy, Vol. 6 No. 5, pp. 32-37, 2008

[30] Nguyen, Q.L., Sood, A., (2011), "A Comparison of Intrusion-Tolerant System Architectures", IEEE Security & Privacy, Vol. 9 Nr. 4, pp. 24-31, 2011

[31] Nguyen, Q.L., Sood, A., (2011), "Designing SCIT Architecture Pattern in a Cloud-based Environment", 41st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), pp. 123-128

[32] Pearce, M., Zeadally, S., Hunt, R., (2013), "Virtualization: Issues, security threats, and solutions", ACM Computing Surveys (CSUR) Journal, Vol. 45 Nr. 2, February 2013, Article No. 17

[33] Pinheiro, E., et.al., (2001), "Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems", Proceedings of the international Workshop on Compilers and Operating Systems for Low Power

[34] Rutkowska J., (2006), Invisiblethings.org, Blue Pill Project

[35] Sabahi, F., (2011), "Virtualization-Level Security in Cloud Computing", IEEE 3rd International Conference on Communication Software and Networks (ICCSN), pp. 250-254, 2011

[36] Schwartz, M., (2012), "New Virtualization Vulnerability Allows Escape To Hypervisor Attacks" http://www.darkreading.com/risk-management/new-virtualization-vulnerability-allows-escape-to-hypervisor-attacks/d/d-id/1104823

[37] SCIT, (2014), http://cs.gmu.edu/~asood/scit/

[38] Smith M., Schridde, C., Freisleben, B., (2008), "Securing stateful grid servers through virtual server rotation", Proceedings of the 17th International Symposium on High-Performance Distributed Computing (HPDC'08), pp. 11-22, ACM 2008

[39] Rashid, F.Y., (2012), "VUPEN Method Breaks Out of Virtual Machine to Attack Hosts", http://www.securityweek.com/vupen-method-breaks-out-virtual-machine-attack-hosts

[40] Williams, D.W., et.al., (2009), "Security through Diversity: Leveraging Virtual Machine Technology", IEEE Security & Privacy, Vol. 7 No. 1, pp. 26-33

[41] Wojtczuk, R., (2008), "Subverting the Xen Hypervisor", http://invisiblethingslab.com/itl/Resources.html

[42] Wojtczuk, R., Rutkowska, J., (2011), "Following the White Rabbit: Software attacks against Intel(R) VT-D technology", Invisible Things Lab, http://invisiblethingslab.com/resources/2011/Software Attacks on Intel VT-d.pdf

[43] Yongki, K., Lim, J., Doo, S., Yoon, H., (2012), "The design of adaptive intrusion tolerant system(ITS) based on historical data", International Conference for Internet Technology And Secured Transactions, pp. 662-667