Copyright ⓒ2017 KSII

VirtAV: an Agentless Runtime Antivirus System for Virtual Machines

Hongwei Tang[1,2,3,4], Shengzhong Feng[1,2,3], Xiaofang Zhao[3,4] and Yan Jin[3,4]
[1] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
Shenzhen, 518055 - China
[e-mail: tanghongwei@ict.ac.cn, sz.feng@siat.ac.cn]
[2] Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences
Shenzhen, 518055 – China
[3] University of Chinese Academy of Sciences
Beijing, 100049 - China
[4] Institute of Computing Technology, Chinese Academy of Sciences
Beijing, 100190 - China
[e-mail: zhaoxf@ict.ac.cn, jinyan@ncic.ac.cn]
*Corresponding author: Hongwei Tang

Received August 9, 2016; revised May 4, 2017; accepted July 10, 2017; published November 30, 2017

Abstract

Antivirus is an important issue to the security of virtual machine (VM). According to where the antivirus system resides, the existing approaches can be categorized into three classes: internal approach, external approach and hybrid approach. However, for the internal approach, it is susceptible to attacks and may cause antivirus storm and rollback vulnerability problems. On the other hand, for the external approach, the antivirus systems built upon virtual machine introspection (VMI) technology cannot find and prohibit viruses promptly. Although the hybrid approach performs virus scanning out of the virtual machine, it is still vulnerable to attacks since it completely depends on the agent and hooks to deliver events in the guest operating system. To solve the aforementioned problems, based on in-memory signature scanning, we propose an agentless runtime antivirus system VirtAV, which scans each piece of binary codes to execute in guest VMs on the VMM side to detect and prevent viruses. As an external approach, VirtAV does not rely on any hooks or agents in the guest OS, and exposes no attack surface to the outside world, so it guarantees the security of itself to the greatest extent. In addition, it solves the antivirus storm problem and the rollback vulnerability problem in virtualization environment. We implemented a prototype based on Qemu/KVM hypervisor and ClamAV antivirus engine. Experimental results demonstrate that VirtAV is able to detect both user-level and kernel-level virus programs inside Windows and Linux guest, no matter whether they are packed or not. From the performance aspect, the overhead of VirtAV on guest performance is acceptable. Especially, VirtAV has little impact on the performance of common desktop applications, such as video playing, web browsing and Microsoft Office series.

Keywords: agentless, antivirus, antivirus storm, virtual machine, virus signature

A preliminary version of this paper appeared in IEEE ICACT 2016, Feb 2-4, Korea. This version includes a detailed description on the basic idea, design and implementation of VirtAV.

https://doi.org/10.3837/tiis.2017.11.026 **ISSN : 1976-7277**

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

1. Introduction

Antivirus protection is necessary for modern computers. As virtualization technology has been extensively adopted to enhance the effectiveness of resource utilization, more and more applications are deployed in VMs. Currently three antivirus approaches, i.e., internal approach, external approach and hybrid approach, are provided in the industry and academia. In the **internal approach**, antivirus software is installed in the VM and it often uses signature-matching [1][2] technology to search viruses in files [3][4]. When all guest VMs on a host machine schedule virus scanning or signature database updating simultaneously, the host will be overloaded and corresponding performance of VMs will be degraded dramatically. As a result, the 'antivirus storm' problem [5] is inevitable because of a large number of concurrent resource-intensive operations stressing on both computing resource and I/O resource. On the other hand, as a useful approach that facilitates VM fault tolerance and system maintenance, snapshot & rollback also has been applied. However, when the VM rollbacks to a snapshot previously saved, the signature database installed inside may be out of date [6][7]. In this case, antivirus software may be unable to detect newer viruses timely, which leads to security vulnerabilities of antivirus peculiar to virtualization environment. We call it as a 'rollback vulnerability' problem. Moreover, antivirus software is highly susceptible to attacks when exposed in the VM under protection [8].

The **external approach** is inspired by the feature that the VMM has supervisory privilege on guest VMs. Antivirus software is totally moved out of the VM [8][9][10], and gets necessary information from the guest OS using VMI techniques [11][40][46]. The existing work following this approach can only detect virus in guest files or memory, and can't prohibit it properly.

The **hybrid approach** is usually composed of two parts: hooks on the guest OS to monitor events and capture information, and scanning engine deployed in a dedicated VM [5][12][13]. The most widely used antivirus product from VMWare and Trend Micro follows this approach [13]. Although this approach can avoid the antivirus storm problem and the rollback vulnerability problem, the hooks or agents in the guest OS are still vulnerable to attacks.

Furthermore, some research shows that, over 80% of modern viruses appear to be using packing techniques to evade detection by antivirus software. There is also evidence that more than 50% of new viruses are generated by simply packing existing ones [48][49]. However, traditional approaches widely adopted by antivirus software is using unpacking routines to recover the original virus program before scanning it [3]. The limitation of this approach is that it needs a specific unpacker for each packer, and moreover, only viruses packed with known packers can be unpacked and detected.

Motivated by the aforementioned observations, an agentless antivirus system VirtAV that is based on in-memory signature scanning is proposed in this paper. Basically, it can be classified into external approach since signature scanning and database updating are performed completely outside of guest VMs under protection. The main contributions are listed as follows.

- We designed an agentless approach for runtime antivirus protection on VMs, which actively scans guest binary codes on the VMM side, transparently performs virus detection and timely prohibits virus found. Such an approach guarantees the security of antivirus system to the greatest extent because there is no attack surface exposed to the

DOCKET ALARM Find authenticated court documents without watermarks at docketalarm.com.

outside world. It avoids the antivirus storm and the rollback vulnerability problems. Moreover, it can also detect packed virus relying on the virus to unpack itself in memory.

- We proposed memory signature of virus, variant of file signature used in traditional antivirus software, which can be used to uniquely identify virus when it has been loaded or even partially loaded into memory. Such an approach helps antivirus system detect and prohibit virus in memory before it is activated.
- We implemented a prototype system based on the Qemu/KVM hypervisor and the open-source scanning engine of ClamAV. Functionality verification reveals that VirtAV is independent of guest operating systems, and is able to detect both non-packed and packed viruses running in user space or kernel space inside guest VMs. Moreover, benchmarks of typical desktop applications, such as Windows booting, video playing and synthetic office workload, are evaluated in both single-VM and multiple-VM environments. The performance results show that the overhead of VirtAV is reasonable and acceptable. From the results, we can further draw a conclusion that antivirus storm is eliminated by VirtAV in the multiple-VM environment.

The rest of the paper is organized in the following manner: Section 2 gives a brief introduction of Qemu/KVM hypervisor and ClamAV antivirus system, and reviews related works on security protection on virtual machines. Section 3 discusses the main design principles on VirtAV. Section 4 gives the implementation details of VirtAV. Section 5 presents the experimental results from the viewpoints of function and performance. Finally Section 6 concludes the paper and mentions the future work.

2. Background and Related Work

2.1 Qemu/KVM Hypervisor

Qemu/KVM hypervisor is a full virtualization solution based on Linux for X86 platform with CPU virtualization extensions (Intel VT or AMD-V). It provides isolated virtualized hardware environment for virtual machines running unmodified Linux or Windows as guest OS. Guest codes (user-level processes or kernel-level OS) run natively on CPU in non-root mode with the exception of sensitive instructions or operations, such as accessing I/O port, which are trapped to KVM, and emulated by KVM and Qemu. For each virtual machine, there is an independent memory address space that starts from physical address 0x0. On the host, EPT (from Intel, or NPT from AMD) is used by MMU to translate guest physical memory address (GPA) into host physical memory address (HPA) when CPU that is operating in non-root mode accesses guest memory. Moreover, if the corresponding host memory page has not been allocated or mapped, or there are not sufficient access rights on the page, EPT violation will be generated and captured by KVM.

2.2 ClamAV

ClamAV is the most widely used open-source antivirus system, which supports detection of both non-polymorphic viruses and polymorphic viruses. For non-polymorphic viruses, signatures are in simple string format, and the Boyer-Moore (BM) algorithm is used to detect this kind of viruses. While for polymorphic viruses, signatures could contains regular expressions and wildcards, and the Aho-Corasick (AC) algorithm [2] is adopted.

Specially, the implementation of the AC algorithm in ClamAV uses a trie to store the finite-state machine (FSM) constructed from the signatures. In addition, 3 helper functions including *goto*, *failure* and *output*, are defined accordingly. The *goto()* function on a given

DOCKET ALARM Find authenticated court documents without watermarks at docketalarm.com.

state points out which state to move if the next input character matches with any predefined input of the state, otherwise, the FSM resorts to the *failure()* function. The *output()* function summarizes and outputs target patterns matched in the end. At the initialization phase, fixed string parts of each polymorphic signatures are loaded from the database, and are used to build the trie. At the scanning phase, ClamAV scans an input file byte-by-byte and detects occurrences of signatures. In general, fixed parts of signatures are scanned in the trie firstly, and after that wildcards and regular expressions are processed. Take a signature with a wildcard as an example. The fixed string is split by an asterisk wildcard into two substrings that is in the "substring1*substring2" format. The two substrings are matched individually by the AC FSM. When all the two parts are found, it will further verify the order and gap between them. ClamAV uses a 256-element array for each node with each element corresponding to an ASCII character. The integer value of the input byte is used as the index into the array, and if the element is present, a match character is found. Furthermore, ClamAV provides several common unpackers, such as MEW, Upack, Petite, yC, FSG, AsPack, WWPack, NsPack and etc., to recover the original programs from packed viruses obfuscated with the known packing algorithms.

2.3 Related Work on Intrusion Protection of Virtual Machines

Virtualization technology brings opportunities for security and is extensively used to improve the anti-attack capability of intrusion or malware detection and protection tools [8][26].

A. VMI-based intrusion protection

Security tools can be installed outside of the VM that need to be protected, and monitors the guest via special interfaces, such as VMI [46]. As is isolated from the VM, the security tools are immune to attacks even if the VM is compromised by attackers. However, as lacking of guest OS-level semantic (known as the semantic gap [40][46]), the accuracy and effectiveness of the security tools outside the VM have to be sacrificed. Livewire [8] is a general IDS framework adopting VMI technology, which uses the priori knowledge about the guest OS data structures to interpret the hardware-level view in OS-level semantics. Especially, to detect signatures of malicious program, it performs a full scan of all the host memory. However, this methodology cannot timely find out virus and prohibit them. VMwatcher [9] reconstructs semantic views of guest OS in a non-intrusive manner, and provides objects with the guest OS view to external malware detection tools. Just like Livewire, VMwatcher only provides malwares detection and cannot intervene execution of the guest. [12] proposed a hybrid approach enabling security tools to perform active monitoring while pertaining isolation to ensure security. It hooks the guest OS to monitor interested events and handle the events in an isolated VM. It solves the semantic gap problem in a simple manner, but it heavily depends on integrity of the guest OS. [20] described a framework that enables security monitoring applications to be placed in the untrusted guest VM without sacrificing the security guarantees. The monitoring application itself is protected by the VMM and should be self-contained to ensure its integrity. This method imposes restrictions on the monitoring applications and even requires reconstruction of commodity tools to adapt to this framework. [27] proposed a mechanism to verify the integrity of user and kernel code at runtime in page granularity. [50] described a framework that combines intrusion monitoring, evidence preservation, in-depth log analysis and decision making on suspicious events handling for guest VMs.

In addition, there are also advances in the research of VMI. VMI technique requires privilege access to the virtual machine monitor, so that it is usually not provided in public

DOCKET ALARM Find authenticated court documents without watermarks at docketalarm.com.

cloud environments. CloudVMI [38] virtualizes the VMI interface and allows cloud users to introspecting their own VMs. For live VM monitoring by VMI, VMI tools and guest VMs run concurrently which might cause race conditions and data inconsistency. TxIntro [47] leverages the atomicity of hardware transactional memory to ensure concurrent and consistent VMI. To reduce the overhead of VMI and improves its generality for different OSes, [43] presents an approach that redirects the system calls of monitoring tools to the dummy process in guest VM which collects guest OS states natively in a lightweight manner. [45] presented the observation that the definition and layout of critical information in kernel data structures for process is stable as the evolution of kernel versions. The generality of VMI is improved by reconstructing the process list basing on only partial information, that is believed as sufficient for intrusion detection. [52] introduced hypervisor introspection (HVI) to detect hypercall-based attacks with hardware supports, such as nested virtualization and EPT protection.

B. Not VMI-based intrusion protection

To solve the antivirus storm problem, VMware & TrendTech [5][13] proposed a so-called agentless antivirus framework - vShield Endpoint. In that framework, the primary causes of storm including signature scanning and database updating are offloaded to a dedicated VM, called secure virtual appliance (SVA). Moreover, there is a lightweight agent (called 'EPsec Thin Agent') deployed in guest OS whose main duties are monitoring file operations in guest OS and forwarding files to SVA to be scanned. However, the framework is not truly agentless, and indeed, the agent is exposed to attacks from malicious users or malware. Moreover, the agent depends on the guest OS. When the agent is not running, for example in booting phase or shutting down phase of the guest OS, the guest VM loses protection of antivirus.

[24] presented an external approach to malware analysis which can hide the analyzer from the target. With the help of hardware virtualization technology such as Intel VT, it can offer both instruction level and system call level tracing of the target in guest VM. [25] described a Qemu-based system that dynamically analyses Windows kernel-level code and extract malicious behaviors from rookits. [37] introduced an event-logging based reliability and security monitoring framework for virtual machines, which relies on hardware invariant to provide an isolated root of trust, so that the events and states about guests cannot be modified by attackers and failures inside guests. [41] presented an out-of-VM user-mode process execution monitoring which supports existing user-mode process monitoring tools such as strace, ltrace and gdb. The suspect process to be monitored is moved out from the production VM to the monitor VM, where the user-mode monitoring tools run side-by-side with the process. This approach removes semantic gap and can directly intercept the process execution at the granularity of user-level function calls. CIVIC [44] creates a live replica of the production VM and the inspection or analysis operations are performed on the replica. It leaves the production VM unmodified without any impact or side-effects during monitoring. HyperCoffer [42] is a hardware-software co-designed framework that guards the privacy and integrity of tenant's VMs in cloud environment. By extending processor virtualization with memory encryption and integrity checking to secure data communication with off-chip memory, it can protect guest VMs against an untrusted hypervisor and physical attacks. [51] described a network-based computer worms detection system for virtualization environment, which runs isolated from the guest VM on the hypervisor. [6] proposed an audit based approach to protect against VM rollback attack which logs all the suspend/resume and migration operation and audits the log for checking malicious rollback behaviours. Furthermore, there are also researches [21][22][23] focusing on malware analysis with the help of virtualization techniques.

DOCKET ALARM Find authenticated court documents without watermarks at docketalarm.com.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.