CERIAS Tech Report 2009-10
Client Honeypots on ReAssure
by Jason D. Ortiz, Pascal Meunier
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Purdue University CERIAS

Client Honeypots on ReAssure

Justification for a Bare-Metal Implementation
CERIAS TR 2009-09

Jason D. Ortiz, Pascal Meunier

Abstract

Client honeypots are typically implemented using some form of virtualization to contain malware encountered by the client machine. However, current virtual environments can be detected in multiple ways by malware. The malware can be executed from within a browser or require escaping from the browser to detect the virtualization. In many cases, detection is accomplished by a simple test. Malware can then modify its behavior based on this information. Thus, an implementation of client honeypots which does not depend on virtualization is needed to fully study malware.

Introduction

Malware is present on hundreds of websites and servers around the world, threatening client-side security. In order to detect and prevent malicious activity, many client-side defenses have been developed, including firewalls, antivirus protection, and client honeypots [5,8,10]. As a result, malicious hackers have attempted to adapt to these defenses by either defeating them, or evading them.

Virtual environments are a popular form of client-side defense. A virtual machine protects a host machine by isolating any compromise which takes place, and in some instances, gives the ability to examine a compromise in greater detail. Nonetheless, attackers have discovered ways to circumvent this defense, such as detecting virtual environments. With this information the attacker could choose to avoid that particular host or to attack the virtual machine monitor (or VMM).

Virtualization-aware malware poses a problem as it may prevent researchers from easily studying the malware, and may require them to run the malware in a non-virtualized environment to observe its effects. Otherwise, as the malware refuses to run in a virtual environment, researchers will not detect a compromise and assume that there was no malware.

Client honeypots are used by researchers to discover client-side attacks. A client honeypot is a mechanism that browses the internet seeking attacks. After an attacker occurs, the client honeypot can contain the malware to allow researchers to examine its behavior. Client honeypots typically use virtualization to contain an attack. Thus, virtualization-aware malware could change its behavior before being detected by a client honeypot.

The above outlines our justification for a bare-metal implementation of a client honeypot. The most effective implementation is to have both a virtual implementation and a non-virtualized or bare-metal implementation. This allows us to be able to detect attacks which operate regardless of a virtualized environment, as well as those which change their behavior when operating within a virtual machine.

The CERIAS project, “ReAssure” provides a test bed of computers which can be reimaged in the case of a compromise or crash. This lends itself well to a bare-metal implementation of a client honeypot. Using “ReAssure”, CERIAS professionals, students and partners could cooperate to observe, analyze, and defend against the growing threat of client-side attacks, especially attacks against virtual environments.

Detection Techniques

There are many techniques used to detect the presence of a virtual environment on a host machine. Some of these techniques are specific to a certain type of virtual machine (such as VMWare or Xen) and others are more general. These detection techniques break down into two categories; non-intrusive and intrusive.

Non-intrusive techniques focus on detecting a VMM (Virtual Machine Monitor) before delivering a malicious payload. In this scenario, when a VMM is detected, the malicious payload is not delivered to the target machine. This type of detection technique is currently not widely used (CERIAS and other professionals have been unable to discover it in the wild) but it still threatens client-side security. Non-intrusive methods typically do not trigger client-side defenses and thus, malware using this technique would remain largely undetected.

A very simple example of a non-intrusive method would be a port scanner implemented in Javascript or Flash. In the case where VMWare is being used with NAT (Network Address Translation), port 5658 will be opened. An attacker could simply check if port 5658 is opened on a guest OS[2]. However, it is worth nothing that this method would not be perfect because other applications could use that port. For example, Gran Turismo 5 Prologue also uses port 5658 as one of many ports used for online races [11]. Still, it is not likely that a target machine would be using port 5658 for Gran Turismo. Thus, an attacker could conclude that if 5658 is opened, the target is using VMWare on NAT [2,11].

Intrusive virtualization detection techniques are quite different from non-intrusive techniques. Malware using an intrusive detection technique would first compromise a target browser. It then executes some type of VMM detection usually allowing the attacker to examine the guest environment for signs of virtualization [2,4,9]. For example, if the attacker finds that the VMWare Tools process is present, then one can assume the host is using VMWare.

Some more sophisticated intrusive methods of detecting virtualized environments include examining local data tables and time frame analysis. Many of these methods have been researched and are well documented for specific virtual environments as well as more general ones [10].

For example, the IDT or Interrupt Descriptor Table can be used to determine if a system is virtualized [6]. The IDT data does not require privileged operating levels to be read. This data could be used to determine the operating environment of a target machine. Both RedPill[7] and ScoopyNG[3] use this method for detecting the presence of virtualized environments [6].

Another example of a complex intrusive technique would be a time analysis. If an attacker knows (or guesses) the hardware configuration of the remote machine, time analysis techniques can be relatively straightforward [1]. An attacker could execute a series of virtualization intense instructions (instructions with a high virtualization overhead) and compare the elapsed time to a benchmark [9].

Moreover, “fuzzy benchmarking” can be used to detect a remote VMM without knowledge of the hardware configuration [1]. The “fuzzy benchmarking” algorithm consists of executing benchmarking code at a high privilege level on the guest OS and analyzing the performance. Although it has not been

shown to work for all configurations, if an attacker has a small amount of knowledge of the target, the success rate of this technique is relatively high [1].

Thus, it is established that current implementations of VMMs can be easily detected. Virtualization-aware malware then has the opportunity to evade any potential client honeypot that is using a VMM. However, evading is not always an optimal strategy for an attacker. As there is a strong push for more virtualization including virtual desktops, malware evading virtualization would become ineffective, because virtual environments would protect users from malware. However, this scenario is not likely to occur for the following reasons.

If malware can detect virtualization, then it can also attack it. An attack could be specifically catered to a virtual environment, and could possibly escape that environment and damage the host system.

In addition, over confident users would perceive a virtual desktop to be completely secure, and immune to attacks. This perception could cause them to be more inquisitive of dangerous websites and less aware of security threats. Any user who believes they are safe is much less likely to be attuned to warnings pointing to the fact they are *not* secure. This gives the attacker the element of surprise. The perception of security (when there is little or none) puts users at a greater risk and attackers will soon be looking to exploit their advantage.

Client Honeypot Implementation

Implementing client honeypots on bare-metal architecture alongside a virtualized implementation could help researchers learn more about attacks against virtualization. It would highlight how the behavior of malware changes depending on the environment it is running in and would allow researchers to observe those differences. Once researchers can begin learning different virtualization attacks, they can begin taking action to prevent these sorts of attacks as well as educating the public of new dangers.

Learning about attacks against virtualization before those attacks begin occurring on a wide scale would be a great step toward better client-side security. It would give researchers and security professionals the opportunity to deploy more secure software implementations.

[1] Franklin, Jason. “Remote Detection of Virtual Machine Monitors with Fuzzy Benchmarking”

[2] “How to detect Virtual PC or VMWare from your program.” http://www.codegurus.be/codegurus/Programming/virtualpc&vmware_en.htm

[3] Klein, Tobias. Trapkit. http://www.trapkit.de/research/vmm/scoopyng/index.html

[4] Microsoft HoneyMonkey Project http://research.microsoft.com/HoneyMonkey/

[5] The Honeynet Project. http://project.honeynet.org

[6] Quist, Danny and Val Smith. “Detecting the Presence of Virtual Machines Using the Local Data Table.”

[7] Rutkowska, Joanna. “Red Pill… or how to detect VMM using (almost) one CPU instruction.” http://www.invisiblethings.org/papers/redpill.html

[8] Wang,Kathy. MITRE Dataset. Powerpoint presentation

[9] Wang, Yi-Min, Doug Beck, Xuxian Jiang, Roussi Roussev, Chad Verbowski, Shuo Chen, and Sam King. “Automated Web Patrol with Strider HoneyMonkeys: Finding Web Sites That Exploit Browser Vulnerabilities.” Microsoft Research. Redmond.

[10] Wolfgarten, von Sebastian. „Virtuelle Leimruten; Honeypots zur Analyse von Angriffen auf Clients und Server“ (This article is in GERMAN).

[11] www.gamespot.com/ps3/driving/granturismo5prologue/show_msgs.php?topic_id=m-1-44553900&pid=942026 (Player Blog, user kennygreen). “Whens that big patch coming out again?”