International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
International Conference on Industrial Automation and Computing (ICIAC- 12-13th April 2014)

RESEARCH ARTICLE OPEN ACCESS

A Novel Approach to System Security Against Split Personality Malware

Nisha Balani
*(Department of Computer Science, Jhulelal Institute of Technology, Nagpur.

ABSTRACT-
The Malware is known as malicious code and malicious software. Malware is fed into the victim's system in order to destroy the victim's data, applications or operating system or otherwise annoying or disrupting the victim. Malware analysts use Virtual Machines to determine any sample program and check them to detect if the program contains any malicious code. In this process if the Malware harms the Virtual Machine, malware developers will simply discard that program and then reload the fresh image of VM to increase system productivity. Malware developers have come up with a new class of Malware called as Analysis Aware Malware or Split Personality Malware (SPM). This malware first detects the presence of Virtual Machines. If Virtual Machine is present, SPM behave like any other normal application, thus escaping the malware detection process. When SPM sample runs on the native machine, it shows its actual behavior and destabilizes the host OS. In presented research work describes the techniques to detect the presence of Virtual Machine and then provides the solution to counter the Split Personality Malware. The solution will be helpful for most of the security analysts to destroy Split Personality malware.
Keywords: Detecting VM, Detecting Virtual Machines, Analysis Aware malware, Split Personality malware, guest OS, host OS.

I. INTRODUCTION

The Impelled by the proliferation of high speed connections and the global coverage, Internet has become a powerful means for knowledge sharing as well as commercialization. The increasing dependence on the Internet, however, also makes it an obvious target for the miscreants to spread computer viruses and other types of malicious software (malware).

The power of malware has reached the level where it can not only penetrate, manipulate and destroy information systems but can even reside on them indefinitely gaining complete control over them without the user getting the slightest hint. This is mainly due to two important factors. Firstly, plentiful vulnerabilities in the operating systems, browsers and other applications are being continuously discovered and exploited by the malware developers prior to any patches being developed against them by the security researchers. Such attacks are termed as zero day attacks. Secondly, the malware developers make use of several obfuscation techniques in order to evade detection by the various anti-malware products especially the ones based on signature detection schemes.

Virtualization, now-a-days serve as a very useful technology in the field of security research and has gained widespread acceptance in the fraternity. It is very popular amongst malware researchers since any suspicious malware samples can be executed on the virtual machines without having the native systems being affected. Since many malware tend to destabilize the host systems, allowing them to run in a virtual environment increases the productivity of the analysts. This decreases the time and cost that the analysts need to study malware behaviors enabling them to build patches against the vulnerabilities that the malware exploit.

However Malware developers have again added an analysis aware functionality to the malwares. SPM analyzes the system in which it is running. If SPM find that it is running on Virtual Machine, it behaves like normal applications hiding their malicious nature and thus escaping detection. When SPM sample runs on the native machine, it shows its actual behavior and destabilizes the host OS.

This paper is divided into 5 sections. Section 1 discusses techniques used for VM Detection. In section 2, Related work done in the area of Split Personality malware is described. In section 3 the research work to detect and defeat Split Personality malware is discussed. In section 4, the implementation details of the solutions VMDetectGuard and WindowsVM is given. In section 5, experimental results are shown. Section 6 gives conclusion and Future scope.

II. VM DETECTION TECHNIQUES

The There are various ways of VM detection, all of which can be classified in one of the following categories:

1.1 Hardware Fingerprinting

In VM there is specific virtualized hardware which is not present in normal machines. This hardware can reveal important information about VM specific components and hence the presence of VM can be detected[1]. Table-1 shows the hardware fingerprinting results on native machine as well as on VMware. This hardware fingerprinting was performed using Windows Management Instrumentation (WMI) classes and APIs.

1.2 Registry Check

There are many references to the string "VMware" in the registry entries of guest OS. Checking the registry values for certain keys clearly reveals the VM presence. The following are a few examples:

HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\Scsi\Scsi Port1\Scsi Bus 0\Target Id 0\Logical Unit Id 0\Identifier
-VMware, VMware Virtual S1.0

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Class\{4D36E968-E325-11CE-BFC1-08002BE10318}\0000\DriverDesc
-VMware SCSI Controller

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Class\{4D36E968-E325-11CE-BFC1-08002BE10318}\0000\ProviderName
-VMware, Inc.

1.3 Memory Check

This technique involves looking at the values of specific memory locations after the execution of instructions such as SIDT (Store Interrupt Descriptor Table), SLDT (Store Local Descriptor Table), SGDT (Store Global Descriptor Table), and STR (Store Task Register)[1][3]-[7]. This technique is used excessively by VM detecting Malware.

1.4 VM Communication Channel Check

A host-guest communication channel is detected using this technique. The IN instruction is a privileged instruction. This instruction raises the exception “EXCEPTION PRIV INSTRUCTION'” when run from a protected mode OS[1][7]. However, when it is running on VMware, no such exception is generated. Instead, VMware initiates guest to host communication by calling the IN instruction. If the magic number (VMXh) is returned to the register, then it is certain that the program is running inside VMware.

1.5 Timing Analysis

An obvious yet rare attack against a Virtual Machine is to check a local time source, such as the "Time Stamp Counter" (TSC). Translation Lookaside Buffers (TLBs) can be explicitly flushed out and then the time to access a new page is determined by reading the TSC before and after the access. This duration can be averaged out over the number of TLBs to be filled. Next, the TLBs are filled with known data by accessing a set of present pages and the time to access a cashed page is determined as before. This value can also be averaged over the number of pages in the TLBs. Now, the CPUID instruction is executed. CPUID is the only VM sensitive instruction which on execution flushes out at least some of the TLBs as a side effect. Now each of the pages that were present in the VM is accessed again. If any of the page's access time matches that of a new page, the presence of a VM is revealed[6][7].

1.6 Process & File Check

There are many VMware specific processes that constantly run in the background such as VMwareUser.exe, vmacthlp.exe, VMwareService.exe, VMwareTray.exe. There also exist some VMware specific files and folders[1][7]. Hence querying for these objects could also serve as a method for VM detection. Though this method could easily be fooled, when combined with other detection techniques, it could obtain more reliable results.

III. RELATED WORK

It has been observed that the amount of work done in this area is not substantial. Only few researchers have provided with the detailed description and solution on this topic. Also, majority of them have focused on study of Split Personality Malware and its detection[9][10][11]. Most of the researchers use VM to detect this kind of Malware and once detected native machines are used to tackle them.

Zhu D. et. al.[10] gives two approaches to counter VM-Aware Malware. The first makes use of dynamic analysis to identify known virtual machine detection techniques. The second method that is proposed by them makes use of dynamic taint tracking to detect any impact caused by the input and changes the execution path of the malware. Although authors claim that this method is effective but only two techniques Memory check and registry check have been addressed of all the existing VM detection techniques. Out of these Registry check method is not very effective. And the authors didn't give any technique to counter of VM detection techniques like Process and file check or timing analysis which are also extensively used by advanced split personality malwares.

The work done by Balzarotti et. al.[11] mentioned an approach in which sample is run on a native machine first where its input and output values exchanged with the system are logged and then the sample is run under a virtual machine where its output values that were logged in the native system are replayed. Then the differences in the sample's behavior are observed. This approach is yet not effective since the sample is run on a native machine first.

Carpenter et al.[12] and Guizani et al.[13] have proposed an approach to trick Malware on host OS. Carpenter et.al.[12] gives two different techniques. The approach tries to destroy the malware by changing the configuration settings of the .vmx file present on the host Remaining VM detection techniques remains unaffected and hence can be used by any malware sample to detect the presence of VM. The work done by Guizani et al.[13] provides an efficient solution for Server-Side Dynamic code analysis. The approach deals with countering Split Personality Malware for Memory Check and VM Communication Channel check. However other techniques are not addressed in the approach.

Kalpa et. al.[7] provides an efficient approach to tackle all VM detection techniques on the virtual machine. The tool that has been designed is made with the help of Pin tool. This research work motivated us to do further research in this area.

IV. PROPOSED METHODOLOGY

1.7 Vm Detection Logic

In order to understand the detection mechanisms deployed by the Split Personality malware, a tool called as DetectVM has been developed. This tool specifically detects the presence of VMware virtual machine by looking for the checks that were described in section 1.

Each of the VM detection techniques is assigned a weight.

• Timing analysis	5
• Registry check	10
• Hardware Fingerprinting	15
• VM Communication channel check	20
• Memory check	20

Based on the weights assigned, the probability of VM presence is calculated as:

VMware Detection Accuracy (%) =
$(((mc_{ob}/mc_{max})*20 + (cc_{ob}/cc_{max})*20 + (hf_{ob}/hf_{max})*15 + (rc_{ob}/rc_{max})*10 + (ta_{ob}/ta_{max})*5) / 70) * 100$

where:

mc_{ob} = Total no. of attribute values for Memory Check obtained on querying the system that confirm VMware Presence

mc_{max} = Total no. of attribute values queried for Memory check

cc_{ob} = Total no. of attribute values for VM Communication Channel Check obtained on querying the system that confirm VMware Presence

cc_{max} = Total no. of attribute values queried for VM Communication Channel Check

hf_{ob} = Total no. of attribute values for Hardware Fingerprinting obtained on querying the system that confirm VMware Presence

hf_{max} = Total no. of attribute values queried for Hardware Fingerprinting

rc_{ob} = Total no. of attribute values for Registry Check obtained on querying the system that confirm VMware Presence

rc_{max} = Total no. of attribute values queried for Registry Check

ta_{ob}= Total no. of attribute values for Timing Analysis obtained on querying the system that confirm VMware Presence

ta_{max}= Total no. of attribute values queried for Timing Analysis

V. COUNTERING VM DETECTION LOGIC

The second objective is to destroy Split Personality malware. This approach will secure the system from Split Personality malware that use Virtual machine detection logic. This objective is divided into two modules:

4.1.1 Converting Virtual Machine to Host Machine

In the first part of implementation the analysis, detection and removal of analysis aware malware is carried out on virtual machine. All the API calls made by the sample under test are checked and its low level information is recorded. Then the sample under test is checked to see if it has tried to access any of the information from which it can detect the presence of VM. If the information matches with any of the monitored set of API calls then tool presented in this research work tries to fool the sample, making it to believe that it is running on host machine.

Algorithm:
Input: The sample that is to be tested as malware

```
1.  List out all the API calls that are used to detect the presence of VM.
2.  Test the malware sample.
3.  With the help of Pin tool, hook into the sample.
4.  Set initial output as "Split Personality Malware not Detected".
5.  while the sample is executing, do the following:
6.  Maintain a list of API calls that are being executed by the sample.
7.  Then compare these results with the list of monitored API calls.
8.  If match is found then
9.  Log the activity in a log file
10. Set the final output as "Split Personality Malware Detected".
11. Fool the running sample by providing fake values.
12. else
13. Do nothing
14. End if
15. End while
```

Output: Split Personality Malware Detected / not Detected

4.1.2 Converting Host Machine to Virtual Machine

The objective is to targets are normal users that do not use any Virtual machine and thus run everything directly on native machine. This approach is basically the reverse of the work presented in module-1, i.e. the analysis, detection and removal of analysis aware malware is carried out at host machine. For any sample that is trying to access low level information, it is provided with fake values making the sample to believe that it is running on Virtual Machine. Since any split personality malware behave normally on VM, it will not affect the host machine.

Algorithm:
Input: Any program on the host machine which can be a split personality malware

```
1. Launch dummy processes which correspond to VM specific processes on the virtual machine, these include vmware.exe, vmact-help.exe, etc.
2. Introduce dummy keys to the Windows Host machine registry which are specific to VM, like SCSI Host, etc. Any process trying to access the registry would find VM specific keys, thus deducing that this is a VM and not a host machine.
3. For timing analysis, introduce multiple threads which will slow down the system thus making it behave as a VM.
```

Output: No split personality malware can attack the host machine as it is behaving like a VM

VI. IMPLEMENTATION

1.8 Part-1:

To counter SPM that employ various VM detecting techniques on virtual machine, a solution has been designed and implemented. This section presents a detailed discussion of the tool that has been implemented called as **VMDetectGuard**. The solution has been implemented in the framework provided by the Pin tool released by Intel Corporation[14]. Pin is a tool for the instrumentation of programs. Pin allows a tool (such as ours) to insert arbitrary code in arbitrary places in the executable. The code is added dynamically while the executable is running.

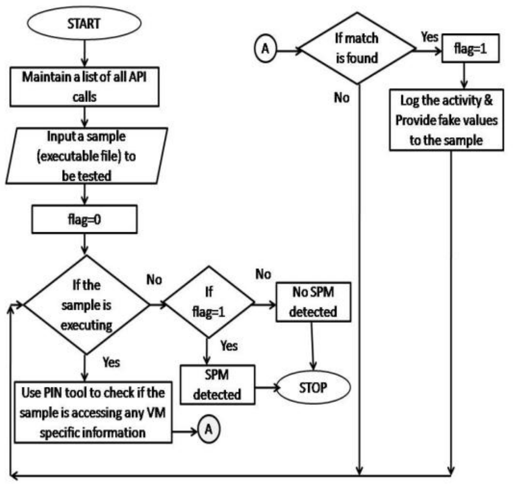

START
Maintain a list of all API calls
Input a sample (executable file) to be tested
flag=0
If the sample is executing
No
Yes
If flag=1
No
Yes
No SPM detected
SPM detected
STOP
Use PIN tool to check if the sample is accessing any VM specific information
A
If match is found
Yes
No
flag=1
Log the activity & Provide fake values to the sample

Figure 1 Flowchart of the development of VMDetectGuard tool

5.1.1 Methodology

As described earlier, all the VM detection methods fall under one or more categories of VM detection. This approach implements the methodology with respect to each VM detection category.

A. Security against Hardware Fingerprinting

In hardware fingerprinting, a list of all API calls that gives hardware information such as Motherboard, BIOS, Processor, Network Adapter etc. is maintained. Then the tool provides false information to the applications that query for such information. Table I summarizes the results.

The tool VMDetectGuard hooks into the sample under analysis and records all the system calls made by it. Whenever any system call matches with the monitored list of API calls, the tool provides fake values to the sample
In Table I the comparative results of Hardware printing is shown. VM returns a value “none” for motherboard serial number. VMDetectGuard returns a more appropriate string such as “.16LV3BS.CN70166983G1XF” instead.

B. Security against Registry Check

The registry querying APIs are monitored by the tool such as RegQueryInfoKeyValue, RegOpenKey, etc. VMDetectGuard intercepts these API calls whenever any of them are executed by the sample. It then looks at the output values returned by the system. If the string such as “VMware”, VMDetectGuard replaces this string with a value that would have been returned on a non virtual system running the same OS.

C. Security against Memory Check

To counter memory check, the presence of SIDT, SLDT and SGDT instructions is detected. This check is currently most popular check in detecting the presence of VM by split personality malware. Whenever any of these mentioned instructions are executed by the malware sample, the tool logs this activity and provides fake values in order to fool the sample.

D. Security against VM Communication Channel Check

The security against VM communication channel check is provided in the same way as security against Memory Check. In this case, the execution of IN instruction is monitored and thus the value of magic number ‘VMXh’ is changed to some other value. This magic number is supplied as an input parameter by the sample under test. Thus “EXCEPTION PRIV INSTRUCTION” exception is generated[7]. This makes the sample to believe that it is not running under VM because on VMware, on executing IN instruction no exception is raised and guest-to-host communication is enabled.

E. Security against Timing Analysis

The instructions such as CPUID and RDTSC (Read time stamp counter) are monitored by the tool. It also keeps the track of number of times each instruction is executed[7]. So, if any instruction is executed repeatedly which exceeds threshold value, for those instructions VMDetectGuard logs the activity. This is done because VM detection through timing analysis makes a single or couple of instructions to execute large number of times since certain instructions when run a large number of times takes considerably much more time on VM than on host machine.

F. Security against File & Process Check

The method used to counter Process and file check is similar to that of Registry checks. If the sample under test makes the request for VMware specific files or processes, the tool sends the ‘File/Process not found’ error.

5.1.2 VMDetectGuard Output

VMDetectGuard provides us with information of presence of Split Personality Malware. It also provides us with the log file that contains the trace of all the VM detection techniques employed by the sample under test. The case where sample under test is not a Split Personality malware, the log file remains empty.

5.2 Part-2

This module is simply the reverse of module-1. This solution is designed and implemented to counter Split Personality malware that employ various VM detection techniques on host machine. In this section, the detailed discussion of the tool WindowsVM is described.

5.2.1 Methodology

In Module-2, the implementation methodology for the following three VM detection techniques is given. This is because the remaining techniques gives results by directly querying the OS and their results cannot be modified. Further research is going on in this area.

A. **Security against Registry Check**

On a virtual machine, normally certain keys are present in the registries that are not present in the host machine. So to defeat the registry check, the similar keys re artificially introduced in the registry of host machine, so that any process which checks for these keys on the host machine will find the artificial keys, thus confirming that the machine as VM and not the host machine.

B. **Security against timing analysis**

On a normal machine, whenever any batch file is copied, it takes less amount of time as compared to that of VM. So to defeat timing analysis test on a host machine, the host machine process is made to slow down. This is done by launching multiple threads which call an infinite loop and thus slowing down the host machine so that it takes considerable time to copy those files.

This makes the sample believe that it is running on a VM.

C. **Security against Process and file check**

Any sample will first check for VM specific processes to identify if it is running inside a host machine or Virtual Machine. To defeat this check, dummy processes are launched that have the same signature as those processes which run inside VM.

5.2.2 WindowsVM Output

For all the above mentioned VM Detection checks, the WindowsVM protects the host machine from being analyzed by any sample making it to believe that it is running in VM and thus protecting our host machine from Split personality Malware. The malware will behave normally since this kind of Malware attacks only on host machine.

VII. RESULT ANALYSIS AND DISCUSSION

Following table-1 summarizes the results of hardware fingerprinting on host machine, on virtual machine when VMDetectGuard is off and on virtual machine when VMDetectGuard is on. The table effectively shows that how our tool can provide fake values to the sample checking for the presence of VM. The values obtained on VM when VMDetectGuard is on are similar to the values that are obtained on host machine thus protecting our machine from Split Personality malware.

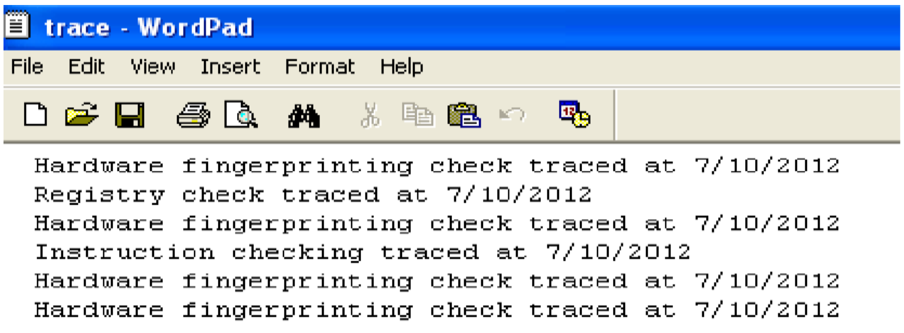
Figure 2 shows the log file that is created whenever any sample tries to check for the presence of VM. The checks that the SPM sample performs are recorded in a text file with the date. This helps to analyze the sample’s behavior.

Table-2 shows the results of the tool WindowsVM on host machine when WindowsVM is off and when WindowsVM is on. This table proves that the tool is capable of converting host machine to VM and thus fooling any sample of Split Personality Malware.

The research done in this area before emphasizes more on Virtual Machine detection. Only few researchers have proposed logic of SPM removal that too on Virtual machine only. The presented research work focuses on defeating SPM on Virtual as well as Host Machine.

VIII. CONCLUSION AND FUTURE SCOPE

The analysis aware malware is increasing day by day and there should be proper measures to defeat them. The developed system(s) work for both for VM and host machines. In the first technique the SPM is fooled to believe that VM is host OS, so it could attack the system and thus let the user know that the program in action is a split personality malware, while the other case was implemented so that if any virus was installed mistakenly in the host machine, then too, it would not affect the system because the Host machine is converted to a virtual machine, and thus securing the system against split personality malwares. The solution provided in this research work is currently built for VMware Virtual machines and Windows OS only. The work could be taken to next level by developing similar methods for other virtual machines and Operating systems.



```
Hardware fingerprinting check traced at 7/10/2012
Registry check traced at 7/10/2012
Hardware fingerprinting check traced at 7/10/2012
Instruction checking traced at 7/10/2012
Hardware fingerprinting check traced at 7/10/2012
Hardware fingerprinting check traced at 7/10/2012
```

Figure 2 Log file created by VMDetectGuard when SPM tries to detect VM present

TABLE 1 COMPARISION RESULTS OF HARDWARE FINGERPRINTING OBTAINED ON NATIVE MACHINE AND VIRTUAL MACHINE

Hardware Component	Attribute Queried	Native Machine	VMware (VMDetectGuard Off)	VMware (VMDetectGuard On)
Motherboard	Serial No.	.5VML2BS.CN701 6689U0M73.	None	.2BTP4BS.CN701 6670MG2DN
Processor	SocketDesignation	Microprocessor	CPU Socket #0	Microprocessor
SCSI Controller	Caption	Microsoft iSCSI Initiator	VMWare SCSI Controller	Microsoft iSCSI Initiator
BIOS	Serial Number	5VML2BS	VMware-56 4d dd 30 4d e2 42 25-43 17 6a 70 65 f3 8a 8a	2BTP4BS
USB Controller	Caption	Intel(R) ICH8 Family USB Universal Host Controller - 2832 Intel(R) ICH8 Family USB Universal Host Controller - 2834	Intel USB Controller	Intel(R) ICH9 Family USB Universal Host Controller - 2932 Intel(R) ICH9 Family USB Universal Host Controller - 2934
Network Adapter	Caption	WAN Miniport (SSTP) WAN Miniport (IKEv2)	VMware Accelerated AMD PCNet Adapter RAS Async Adapter	WAN Miniport (SSTP) WAN Miniport (IKEv2)

TABLE 2 COMPARISION OF RESULTS OBTAINED BY WindowsVM ON HOST MACHINE WITH SECURITY ON/OFF

Check description	Security activated	Security inactive
Checking for VM specific processes like vmhost, vmact-help, etc.	Present	Not present
Checking for VM specific registry keys like SCSI Host, etc.	Present	Not Present
Time required to copy a file of 10000 bytes	Around 10 seconds	Around 3 seconds

REFERENCES

[1] Liston T. and Skoudis E. (2006). “On the Cutting Edge: Thwarting Virtual Machine Detec-tion” [Online]. Available: http://handlers.sans.org/tliston/ThwartingVMDetection_Liston_Skoudis.pdf (Nov 1, 2010)

[2] WMI Classes (2008). “WMI Classes Windows” [Online]. Available: http://msdn.microsoft.com/en-us/library/aa394554%28v=vs.85%29.aspx (Dec 30, 2010)

[3] Quist D. and Smith V. (2005). “Detecting the Presence of Virtual Machines Using the Local Data Table” [Online] Available: http://www.offensivecomputing.net/files/active/0/vm.pdf (Nov 14, 2010)

[4] Vishnani K., Pais A. and Mohandas R. (2011), “Introduction to Malware” Available: http://securityresearch.in/index.php/2011/01/projects/malware_lab/introduction-to-Malware/?ubiquitous_id=17 (Jan 20, 2011) s

[5] Omella A. (2006). “Methods for Virtual Machine Detection” [Online]. Available: http://www.s21sec.com (Nov 24, 2010)

[6] Ferrie P. “Attacks on Virtual Machines”. In the Proceedings of the Association of Anti-Virus Asia Researcher Conference, 2007.

[7] Vishnani K., Pais A., Mohandas R. (Aug, 2011), “Detecting & Defeating Split Personality Malware”. In the Proceeding of Fifth International Conference on Emerging Security Information, Systems and Technologies. France, August 2011.

[8] VMware (2010), “VMware KB: Changing a MAC address in a Windows virtual machine” [Online]. Available: http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1008473 (Jan 15, 2010)

[9] Lau B. and Svajcer V. “Measuring virtual machine detection in malware using DSD tracer”. In the Proceedings of Virus Bulletin, 2008, pp. 181-195.

[10] Zhu D. and Chin E. (2007). “Detection of VM-Aware Malware” [Online]. Available: http://radlab.cs.berkeley.edu/w/uploads/3/3d/Detecting_VM_Aware_Malware.pdf (Dec 10, 2010)

[11] Balzarotti D., Cova M., Karlberger C., Kruegel C., Kirda E., and Vigna G. “Efficient Detection of Split Personalities in Malware”. In the Proceedings of 17th Annual Network and Distributed System Security Symposium (NDSS 2010), Feb 2010

[12] Carpenter M., Liston T., and Skoudis E. "Hiding Virtualization from Attackers and Malware". IEEE Security and Privacy, June 2007, pp. 62-65.

[13] Guizani, W., Marion J.-Y., and Reynaud-Plantey D. “Server-Side Dynamic Code Analysis”. Analysis, 2009

[14] Pin (2004). “Pin - A Dynamic Binary Instrumentation Tool” [Online]. Available: http://www.pintool.org/ (Jan 10, 2010)

[15] VmDetect (2005), “Detect if your program is running inside a Virtual Machine - CodeProject” [Online]. Available: http://www.codeproject.com/KB/system/VmDetect.aspx (Jan 4, 2010)