NOVATEUR PUBLICATIONS
INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY
[IJIERT] ISSN: 2394-3696 Website: ijiert.org
VOLUME 8, ISSUE 7, July. -2021

A REVIEW ON REMOTE VIRTUAL MACHINE DETECTION AND MASKING FROM THE BROWSER ECOSYSTEM

Akhilesh Adithya
BITS Pilani, K K Birla Goa campus, India
akhileshadithya311@gmail.com

ABSTRACT

With the development of computer science, the detection of virtual machines and the masking of the virtual machine monitor is an important topic in academia. However, limited research has been conducted in the field of remote detection of virtual machines. Remote virtual machine detection is used in lots of fields such as cheating detection, sandboxing, and in phishing websites. In this paper, we review the existing methods of detection of virtual machines remotely from with the access limited to the browser ecosystem of the client side. We review 7 different attacks and verify their usability in different setups comprising combinations of multiple host OS and browser ecosystems. We also provide methods to hide the presence of virtual machines in these particular attack strategies. The goal of this paper is to encourage interest in this field.

INTRODUCTION:

With the development of computer science, virtual machines have become a major research topic[10][17], and are being used in multiple fields such as cloud data centers[2], testing malware[7], data recovery, and simulating devices[3][11].

Security engineers utilize virtual machines extensively by creating honeypot environments to test out malicious samples[18]. Hence malware developers actively foil this by attempting to detect the virtual machine environments. The academic community has extensively explored various methods of virtual machine detection and it's prevention[9][20][21].

However, limited research has been conducted in the field of remote detection of virtual machines[15]. Remote detection of virtual machines plays a significant role in applications ranging from preemptive defense of malware to the anti-cheating protocols found in online examinations[19][22]. Phishing sites have been found to detect virtual machines to prevent their detection[5]

This paper reviews the detection of virtual machines remotely with the access limited to the browser. We also suggest possible ways to counter the mentioned virtual machine detection techniques.

In section 2, we provide a taxonomy of related keywords and terminologies. This section also briefs the technologies and tools used in virtual machine detection. Section 3 delves into the various virtual machine detection methodologies and the suggested measures to prevent them with the scope being restricted to remote access from browsers. Section 4 concludes the paper.

TERMINOLOGIES AND CONCEPTS

Before discussing in detail about the virtual machine detection techniques and their countermeasures, in this section we will provide a qualitative taxonomy on different terms and keywords related to virtual machine detection used in this paper.

2.1 Keywords and Definitions

Virtual Machine Monitors (VMM) - VMMs also known as hypervisor, is a software program that sits between a physical host machine and a virtualized one, and allows the creation, management, and monitoring of one or more virtual machines(VM).

Virtual Machine Detection - Virtual machine detection can be simplified as the process of detecting the presence of the VMM from within the virtual machine. There are 4 major methods of detecting VMs[18] -

- Looking for VME artifacts in processes, file system, and/or registry
- Looking for VME artifacts in memory
- Looking for VME-specific virtual hardware
- Looking for VME-specific processor instructions and capabilities

Browser Ecosystem - In the context of this paper, a browser ecosystem would refer to the browser engine [comprising the layout engine, rendering engine, and the javascript engine], the browser interface [comprising the UI elements such as address bar, buttons], and the various addons installed

2.2 Virtual Machine Detection remotely from the browser ecosystem

Standard techniques of virtual machine detection are not applicable in the context of remote access of the browser ecosystem as access to system level resources such as processes, registry, and memory are not allowed. This poses a new challenge in figuring out if the browser is running in a virtualized environment. We classify all methods into 2 major categories through which virtual machines can be detected

2.2.1 Javascript based attacks

Javascript is a client side scripting language that is one of the backbones of the modern internet with 97.4% of all websites utilising it[24]. While utilised mainly in the web domain, it is also used in server sided websites and non browser applications[1][6]. The detection of virtual machines done through pure javascript is the more commonly found one as it can easily be traced and blocked easily.

2.2.2 Non Javascript attacks

Various other libraries and software technologies can help in the detection of virtual machines remotely. Libraries such as openGL, .NET, and even java applets can be used to check for the presence of a VMM. These often use convoluted methods of finding out if the browser is running on a virtualized environment and are prone to false positives.

STUDIES & FINDINGS

In this experiment, we used virtualbox as the VMM. The host OS was 64 bit arch linux. The guest OS used were Windows 10 64-bit and Ubuntu 20.04 64-bit. The VMM was tested on a device with an intel processor, nvidia graphics, and 16GB RAM. The browser ecosystems used were Firefox and Chromium as all modern browsers are a derivative of these two systems.

3.1 WebGL renderer

WebGL is a low level 3D graphics javascript API based on openGL. WebGL can be used to find out the vendor and the renderer which can effectively expose the presence of a VMM[4].

3.1.1 Virtual Machine Detection

VirtualBox utilizes VBoxVGA by default on virtual machines running Windows as the guest OS. WebGL automatically recognises this as the “virtual box graphics adapter”, which can easily lead to detection of virtual machines. This test fails to work on VMs running linux as the VMs utilize VMSVGA

3.1.2 Virtual Machine Masking

As this test relies on VBoxVGA, the VMM display settings can be changed to use VMSVGA. VMSVGA reports CPU based implementations.

3.2 Screen size and aspect ratio

Virtual machines that aren't configured properly tend to have non-standard screen sizes and aspect ratios.[4]

3.2.1 Virtual Machine Detection

The javascript Screen object can be used to find out the screen width, height, and the aspect ratio. The availWidth coupled with devicePixelRatio can be used to calculate the aspect ratio. Non standard aspect ratios or abnormally low screen sizes can be used to identify virtual machines.

3.2.2 Virtual Machine Masking

VMM detection can easily be thwarted by changing the display settings to match a standard screen size[23].

3.3 RAM Size

The size of the RAM available to the virtual machine can be utilised to check for the presence of a virtualized environment. Most VMs have small RAM size as they need to share it with the host OS[4].

3.3.1 Virtual Machine Detection

The navigator API's deviceMemory can be used to figure out the RAM size. Any modern computer is equipped with at least 4 gigabytes of RAM. Any browser running on 2GB of RAM or less can be blacklisted.

3.3.2 Virtual Machine Masking

This can be bypassed by allocating at least 4GB of RAM for the virtual machines. Another possibility is using Firefox as only chromium based browsers support the deviceMemory API

3.4 Estimating number of cores

The total number of cores can be estimated using the hardwareConcurrency protocol from the navigator API[12].

3.4.1 Virtual Machine Detection

Virtual machines have limited access to the total number of cores available as the cores are shared between the host os and the guest os. Any browser that reports only 1 core can be classified as a virtual machine.

3.4.2 Virtual Machine Masking

The number of cores available to the virtual machine can be increased by editing the processor settings in the system tab.

3.5 MAC Addresses

A MAC address is a unique identifier used to identify components in a network address, and is managed by the IEEE organisation[14]. This can be used to identify virtual machines[13].

3.5.1 Virtual Machine Detection

Libraries like ASP.NET or Java applets can be used to find out the MAC addresses. VMMs tend to use mac addresses in certain buckets. This can be exploited to detect the presence of VMMs

3.5.2 Virtual Machine Masking

MAC addresses can be spoofed[8]. Spoofing is a technique for changing the factory assigned MAC address by masking the old identity.

3.6 Performance Object

The javascript performance object's now method can be used to find out the integral number of the window's performance ticks. With multiple samples of this, the underlying time unit can be figured out[16].

3.6.1 Virtual Machine Detection

Virtual machines utilizing chromium based browsers can be detected when using a windows host os. Values of 10000000 Hz (synthetic HPET-based counter) and 3579545 Hz (ACPI-based counter) are rarely used in physical machines but typically used in virtual machines[16]

3.6.2 Virtual Machine Masking

This can easily be prevented by utilising a firefox based browser, or by enabling anti fingerprinting services available in chrome flags. It can also be prevented by using linux as the host OS as it returns a value of infinity.

3.7 Plug and Play devices

Plug and play devices include but are not limited to microphones, USBs, audio cards, web cameras, etc. These devices by default have the name "virtual" attached to them, and hence can be used to detect the presence of virtual machines.

3.7.1 Virtual Machine Detection

The enumerateDevices method from the mediaDevices object of the navigator API can be used to identify the presence of virtual machines. These devices often have the name "virtual" or "vm" in them if they're running in a virtualized environment.

3.7.2 Virtual Machine Masking

This method can be bypassed by renaming the devices in the windows device manager or changing the /dev/inputs and /dev/video files.

CONCLUSIONS

Virtual machines provide immense benefits due to the wide range of applications that they represent. This has made VMs widely accepted in both academia and the industry. Some of the applications such as sandboxing and malware detection require hiding the presence of the VMM, whereas others such as detection of cheating in examinations requires the detection of VMs. This never ending struggle has led to the immense improvement in the standards of both VM detection and its prevention

This concept can be similarly applied in the case of remote detection of virtual machines from the browser environment. In this paper we first identify the categories of virtual machine detection and throw light on the taxonomy of different terms and keywords used. This is done to ensure easier expansion in the future if the

need arises. We also discuss and list out 7 different types of attacks that can be utilised in the detection of virtual machines, and also the ways to prevent the aforementioned detection.

This paper intends to work as a stepping stone to introduce the remote detection of virtual machines from the browser as this field has remained vastly unexplored in academia.

REFERENCES

1) Adobe, "Acrobat SDK 2021". [online] Available at: https://opensource.adobe.com/dc-acrobat-sdk-docs/acrobatsdk/ [Accessed 7 Jul. 2021].
2) Alashaikh A., Alanazi E., and Al-Fuqaha A. (2021). "A Survey on the Use of Preferences for Virtual Machine Placement in Cloud Data Centers". ACM Comput. Surv. 54, 5, Article 96 (June 2021), 39 pages. https://doi.org/10.1145/3450517
3) Android Developers. "Run apps on the fAndroid Emulator." [online] Available at: https://developer.android.com/studio/run/emulator.
4) bannedit (2019), "Virtual Machine Detection In The Browser". [online] bannedit's musings, Available at: https://bannedit.github.io/Virtual-Machine-Detection-In-The-Browser.html [Accessed 7 Jul. 2021].
5) BleepingComputer, "Phishing sites now detect virtual machines to bypass detection." [online] Available at: https://www.bleepingcomputer.com/news/security/phishing-sites-now-detect-virtual-machines-to-bypass-detection/ [Accessed 7 Jul. 2021].
6) Chhetri N. (2016) "A Comparative Analysis of Node.js (Server-Side JavaScript)" .Culminating Projects in Computer Science andInformation Technology. 5. https://repository.stcloudstate.edu/csit_etds/5
7) Cruz J.C.F. and Atkison T. (2011). "Digital forensics on a virtual machine." In Proceedings of the 49th Annual Southeast Regional Conference (ACM-SE '11). Association for Computing Machinery, New York, NY, USA, 326–327. https://doi.org/10.1145/2016039.2016130
8) feross.org - "Spoof your MAC address in Mac OS X" [online] feross.org. Available at: https://feross.org/spoofmac/ [Accessed 7 Jul. 2021].
9) Franklin J., Luk M., McCune J.M., Seshadri A., Perrig A., and Doorn L.V. "Towards Sound Detection of Virtual Machines." [online]. Available at: http://www.cs.cmu.edu/ ~jfrankli/book_ chapters/virtual_ machine_detection.pdf [Accessed 7 Jul. 2021].
10) Goldberg R.P. (1974). "Survey of virtual machine research," in Computer, vol. 7, no. 6, pp. 34-45, June https://doi.org/10.1109/MC.1974.6323581.
11) Gong, Wen-chao, Yongchao Tao, W. Xiang and Xianghu Wu (2018). "Research on Virtual Device Management in Real-time Environment." . https://doi.org/10.2991/ANIT-17.2018.16
12) Grey .E., "CPU core estimation with JavaScript". [online] Available at: https://eligrey.com/blog/cpu-core-estimation-with-javascript/ [Accessed 7 Jul. 2021].
13) Grossman, J. (2009) "Web pages Detecting Virtualized Browsers and other tricks". [online] Available at: https://blog.jeremiahgrossman.com/2009/08/web-pages-detecting-virtualized.html [Accessed 7 Jul. 2021].
14) IEEE Registration Authority, "IEEE SA." [online] Available at: https://standards.ieee.org/faqs/regauth.html#20 [Accessed 7 Jul. 2021].
15) Jämthagen C., Hell M., Smeets B. (2012) "A Technique for Remote Detection of Certain Virtual Machine Monitors." In: Chen L., Yung M., Zhu L. (eds) Trusted Systems. INTRUST 2011. Lecture Notes in Computer Science, vol 7222. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-32298-3_9
16) Klein A., "Web-based virtual machine detection using the HTML5 Performance object, Amit Klein's security corner." [online] Available at: http://www.securitygalore.com/site3/vmd1-advisory [Accessed 7 Jul. 2021].

17) Li Yunfa, Li Wanqing and Jiang Congfeng. (2010). "A Survey of Virtual Machine System: Current Technology and Future Trends". Electronic Commerce and Security, International Symposium. 332-336. https://doi.org/10.1109/ISECS.2010.80
18) Liston T., Skoudis E. (1974) . "On the Cutting Edge: Thwarting Virtual Machine Detection. " [online]. Available at: https://handlers.sans.org/tliston/ThwartingVMDetection_Liston_Skoudis.pdf[Accessed 7 Jul. 2021]
19) Mercer Mettl, "Mettl proctoring suite Security features". [online]. Available at: https://mettl.com/downloads/wp-content/uploads/sites/12/2020/05/MM_Security-Document_V2.pdf [Accessed 7 Jul. 2021].
20) Payne B.D., Carbone M.D.P.D.A., and Lee W. (2007) "Secure and Flexible Monitoring of Virtual Machines," Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007), 2007, pp. 385-397, https://doi.org/10.1109/ACSAC.2007.10
21) Rin N. (2013). "VMDE Virtual Machines Detection Enhanced." [online]. Available at: https://www.heise.de/security/downloads/07/1/1/8/3/5/5/9/vmde.pdf [Accessed 7 Jul. 2021].
22) Safe Exam Browser, "Safe Exam Browser - About." [online] Available at: https://safeexambrowser.org/about_overview_en.html#features-common-features [Accessed 7 Jul. 2021].
23) StatCounter Global Stats. "Desktop Screen Resolution Stats Worldwide." [online] Available at: https://gs.statcounter.com/screen-resolution-stats/desktop/worldwide/#monthly-202006-202006-bar [Accessed 7 Jul. 2021].
24) W3Techs (2020), "Usage Statistics of JavaScript as Client-side Programming Language on Websites". [online] Available at: https://w3techs.com/technologies/details/cp-javascript/ [Accessed 7 Jul. 2021].