EternalBlue, BlueKeep, and DirtyCOW

legacv
December 7, 2023

I. Abstract

Threats to security lie in every component of computers and computing networks – from applications and software down to operating systems and hardware. This paper seeks to define a few notable exploits discovered in the past ten years that attack either kernel-level processes or protocols built into an operating system, including EternalBlue (and its use in the WannaCry ransomware attack), BlueKeep, and DirtyCOW. It also touches on developed countermeasures to those exploits and attempts to lay out best practices for future exploits that may not have a patch in place.

II. Introduction

New operating system vulnerabilities are being discovered weekly; if not by internal researchers, then by external threat actors, who may or may not push malicious zero-day attacks to the world. This paper aims to coalesce knowledge about a few recent protocol-level or kernel-level exploits on Windows and Linux systems (respectively) and expose the damage done when one of those exploits gets turned into self-propagating ransomware. The WannaCry ransomware enabled by the Windows EternalBlue exploit will be explored, along with the Linux DirtyCOW vulnerability and the Windows BlueKeep vulnerability. Terms to know include: threat, meaning a potential risk to an entity; exploit, meaning a program or script that abuses a misconfiguration to endanger the confidentiality, integrity, or availability of a system; vulnerability, meaning a misconfiguration that is able to be exploited; protocol, meaning a set of data transmission rules common amongst all devices that support said protocol; remote code execution, meaning when an attacker is able to execute arbitrary code on a system that is not owned by them; zero-day, meaning an exploit that is current and has no known patches; kernel, meaning base-level operating system code that connects hardware and software components; and worm, meaning an exploit that can propagate itself to other devices and networks.

III. Background and Motivation

This research was inspired by the current threat landscape facing both U.S. and global industries, and aims to highlight just how destructive, unpredictable, and unprecedented targeted operating system (OS) attacks can be. Even through OS providers' best efforts to keep their product patched, maintained, and secure, user error through misconfiguration and lack of updating can expose hundreds of thousands of machines to worms, ransomware, and other threats not covered in this research. Vulnerabilities that are often coupled or confused with OS vulnerabilities can include things like exploits of common libraries, like the Apache Log4j vulnerability, or attacks on hardware that exploit side-channels, like Meltdown and Spectre. However, these are all distinct categories separate from OS attacks. OS attacks include attacks on functionalities built into operating systems themselves, like protocols specific to Windows (SMB and RDP in this case), or attacks on the kernel, the process at the heart of the operating system that manages interactions between hardware and software. They can also include attacks on the code of the operating system itself, but exploits that use that are not covered in this paper.

IV. Vulnerabilities and Exploits

i. WannaCry first emerged in May of 2017 and is reported to have affected more than 200,000 devices across 150+ countries (*What was the WannaCry ransomware attack?*). It is a type of ransomware: malicious code that encrypts all files on a device with an uncrackable key, then forces the device owner to pay a certain sum (typically in cryptocurrency) to an address in order to retrieve the key to decrypt their files. In some cases, ransomware will double as scareware to threaten its victims into paying its price, claiming that the attackers have proof of the victim committing illegal acts like selling drugs (Kumar, 2018). The reason WannaCry was

able to affect so many devices is that it was self-propagating; when it successfully installed itself onto one computer, through a user clicking a malicious link or some such, it then scanned the local area network to see if there were any other vulnerable machines it could copy itself onto (Kumar, 2018).

WannaCry worked off of the EternalBlue exploit (CVE-2017-0144), which exposed a flaw in the Windows Server Message Block (SMB) protocol. SMB protocol, split into versions 1, 2, and more recently version 3, is a protocol built into Windows operating systems that allows shared access to network resources, such as printers, folders, files, and serial ports (Sheldon, n.d.). Though primarily used for Windows computers, macOS and some Linux systems provide built-in support for SMB, given Windows' prevalence in the consumer market. EternalBlue, however, based off known reporting, only affected Windows systems with ports 139 and 445 open, which SMB used. It was an exploit initially developed by the National Security Agency, then leaked through the Shadow Brokers group (Burdova, 2020).

There are three bugs that EternalBlue exploits to obtain remote code execution (RCE). Two allow for a buffer overflow to take place, where the exploit gives more information to a buffer than it is allocated to keep; the third allows for code to be written in the overflowing memory space, which is then executed by the system (SentinelOne, 2019). So, when an attacker creates and sends to a device an SMB packet exploiting this buffer overflow and executing malicious code, the device is now compromised.

WannaCry executes the EternalBlue exploit to compromise a computer, but takes it to an endpoint rather than nebulous arbitrary code. It imports a cryptography API, generates identifiers, scans the filesystem to find all files, generates an AES key per file, encrypts those AES keys and saves them to the files, writes WANNACRY to the files and encrypts them, then "cleans up" and displays the image notifying the victim of the ransomware (Chen, 2017). Another version of this process can be seen in Figure 1. There are steps in-between these that rely on knowledge of specific Windows processes and filesystem hierarchies that are out of scope of this paper, though the references used go in-depth on them.

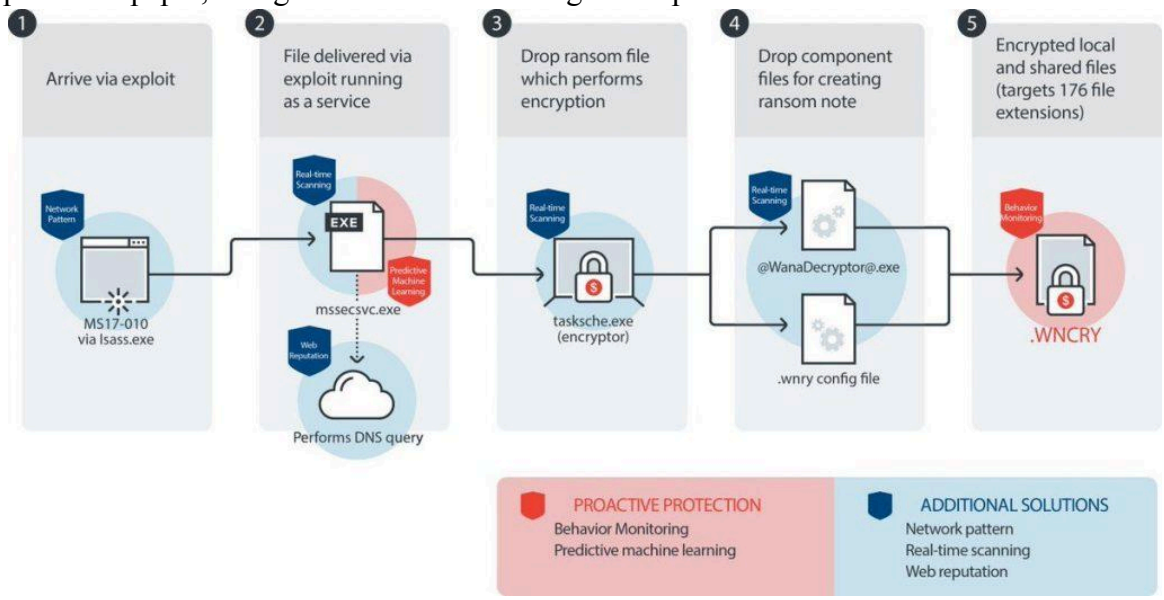

Figure 1: A flowchart of the WannaCry ransomware process. Taken from Natural Networks, https://www.naturalnetworks.com/wannacry-encryption-malware-attack/.

ii. BlueKeep (CVE-2019-0708) is another exploit that affects a protocol built into Windows operating systems. Remote Desktop Protocol (RDP) is a protocol that functions for Windows like Secure Shell (SSH) does for Linux – it allows remote connections to a desktop. RDP, unlike SSH, has a GUI that allows for viewing of the desktop rather than simply a command line, and it runs on port 3389.

It sets up 32 static virtual channels, with dynamic virtual channels being contained in one of those static channels (CertX, 2020). Virtual channels are set up by the core RDP, but are used for "extensions" for RDP, such as "support for special types of hardware, audio, or other additions to the core functionality" (QuinnRadich, 2019). There is a channel named MS_T120 bound to channel 31 that is hardcoded and used internally within the RDP service; if an attacker, pre-authentication, designates another new channel as MS_T120 and binds it to a channel that is not channel 31, they cause heap memory corruption, which can then be manipulated to put unauthorized code into memory and therefore perform remote code execution (Van Impe, 2019).

There is no catastrophic worm such as WannaCry that used BlueKeep; though it is wormable due to the nature of the protocol that it exploits, the only notable deployment of it on more than an individual scale is through an attempted cryptocurrency mining operation. Through honeypots, intentionally vulnerable machines set up by researchers to assess the current threat climate, researchers found that a group was attempting to use BlueKeep to install a cryptocurrency miner on victim machines (Greenberg, 2019). However, it seemed that the miner was not self-propagating, and that the attacking group had scanned for vulnerable machines before targeting them, meaning that the scope of the attack was severely narrowed. In addition, it seemed that the exploit did not always work correctly, and was at risk of crashing target machines occasionally (Greenberg, 2019). Though less of a global threat than EternalBlue and WannaCry, BlueKeep is often mentioned in relation to the two due to its aforementioned wormable nature and its exploitation of a built-in Windows protocol.

iii. Pivoting from Windows and Windows protocols, DirtyCOW (short for Dirty Copy-on-Write) (CVE-2016-5195) is a Linux kernel exploit that allows for local privilege escalation. Meaning, when an unauthorized user exploits it on a Linux system, they are afforded all of the privileges and rights of an administrator on the system. As evidenced in its name, it exploits the Copy-on-Write mechanism in the Linux kernel, which is a resource management technique. When multiple processes ask for the same or indistinguishable resources, all are pointed to the same resources in memory; it is only when a process makes a change to the resource that a copy of the resource is created for the process to make edits to. This saves time especially when a process ends up not modifying a resource (Hare, 2009). It also saves memory space from being wasted on identical copies of a resource.

In affected Linux versions, there existed a race condition in private memory spaces where the COW mechanism was deployed. A race condition occurs when two or more threads attempt to modify a given memory address at the same time, and it is unknown or uncertain which will modify it first. The private memory spaces were intended to be read-only, but due to the timing between the private writable copy of the memory space being created and it being written to, a third call can quickly be executed that deletes the private writable copy and instead redirects the write function to the original private memory space (Wilson, n.d.). The three functions involved are write(), which writes to a file; mmap(), which creates the mapping in memory of the current working file; and madvise(), which points the kernel to the needed address range (Pearson, 2021). Figure 2 shows a flowchart of the DirtyCOW exploit process.

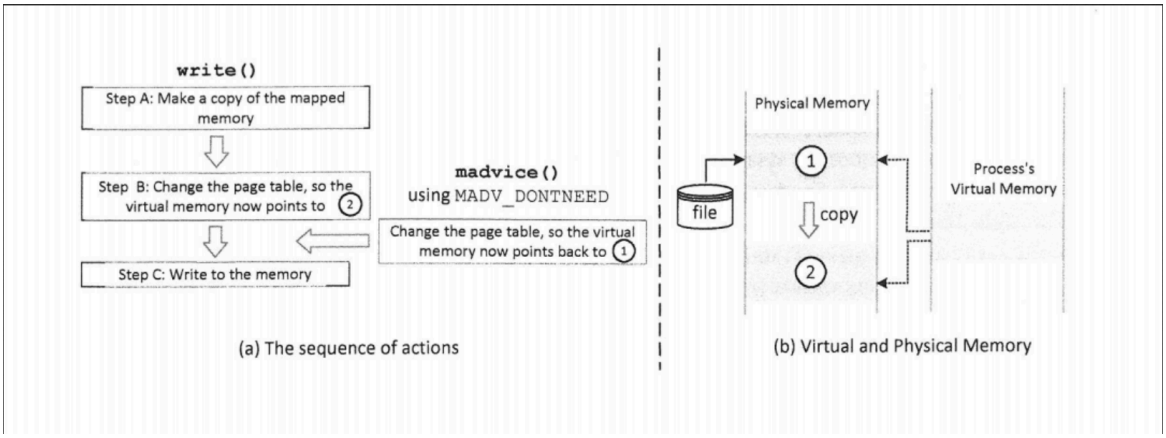

Figure 2: The DirtyCOW exploit process. Taken from https://programmer.help/blogs/dirty_cow_race_condition_attack.html.

Because the nature of the exploit requires an attacker to already have access to a machine, it is not wormable like EternalBlue or BlueKeep. Rather, it is for attackers who have already gained shell or command-line access through other exploits to write to private files such as /etc/passwd or /etc/shadow in order to elevate their privileges. With this exploit, access is as good as unrestricted for the attacker; anything they cannot do with the exploit, they can do with the administrative access they can gain from this exploit. Because it is kernel-level, all distributions (of vulnerable versions) of Linux are affected, because they all use the same Linux kernel to schedule tasks and etc.

V. Countermeasures

A strong solution to many known vulnerabilities and exploits is to keep all operating systems up-to-date with the latest patches and security fixes. However, that is not always possible if a certain exploit is a zero-day with no known solutions.

About a month before Shadow Brokers leaked EternalBlue, Microsoft pushed a security bulletin for it named MS17-010. Installing it would have been the effective solution against EternalBlue and WannaCry; however, not every Windows user installed the patch, leading to over 200,000 device compromises. Also, disabling SMB on computers that did not have a use for it could have prevented a handful of compromises. The day WannaCry spread across the globe, it was halted in its execution by a researcher named Marcus Hutchins. He had reverse engineered the malware and discovered that it queried a nonexistent domain, iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com, before executing any of its ransomware functionality. He registered the domain and found that, once it started receiving a response from the domain, WannaCry would not execute any of its ransomware functionality (*What was the WannaCry ransomware attack?*). It is theorized that the query to a nonexistent domain was a type of sandbox check performed by the malware; sandboxes, or network-isolated environments to test uncertain executables in, try to replicate a full environment to convince the malware to execute. Because of that, they would return a false positive response to the nonexistent domain query. When it received that positive response, WannaCry would "know" it was in a sandbox and not execute so as to prevent reverse engineering. However, when Hutchins registered the domain, every new instance of WannaCry across the globe would be receiving a positive response from the domain, and so prevent itself from executing. That was the post-deployment "killswitch" that was found for WannaCry.

No such killswitch exists for BlueKeep. In May of 2019, Microsoft released security patches for all supported versions of Windows, and even some that were not supported for updates anymore, hinting at the severity of such a vulnerability. Enabling Network-Level Authentication (NLA) for RDP can also drastically decrease the chance of being affected by BlueKeep, as devices with NLA enabled filter who is able to send RDP packets to the device, adding an extra component to the exploitation process. The wormable version of BlueKeep worked off of RDP sessions pre-authentication, which is why it was able to propagate quickly. Also, disabling RDP on devices that have no use for it (as was automatically the case with Windows versions post-BlueKeep) can eliminate the chance of being affected by it, or any of the numerous smaller exploits and bugs linked to it.

The fix for DirtyCOW is even simpler. Because it affects the kernel, it is not a process which can be shut down, nor can it be worked around through other processes. The only countermeasure that can be taken against DirtyCOW is to update the Linux kernel version provided by the developers. Indeed, because many distributions of Linux are open-source, one could theoretically work on fixing the bug and pushing the patch themselves; though, that is best left to professionals, and too time- and resource-intensive for the average consumer.

VI. Conclusion

It's important to both keep all OS versions up-to-date with the latest patches and to ensure that all services being implemented on a device are strictly necessary in order to mitigate the risk of being affected by a worm or other exploit. Staying up-to-date with the latest information on recent security threats is helpful towards proactive defense rather than reactive, which again decreases the risk of being affected by similar exploits. Audits of what software is installed on a machine, even if it is a client machine, and whether that software is necessary and up-to-date can mean the difference between the loss of money and data or a safe operating environment.

The exploits referenced here are not the only ones of their kind. Exploits often come in packs, with one "most notable" exploit chosen and named (such as EternalBlue and BlueKeep) and numerous others found at the same time or later published alongside them affecting the same service. This, again, highlights the importance of maintaining a strong patch schedule and review of any extraneous OS services.

Both professional and open-source OS developers are not invulnerable. All programmers make errors or buggy code, and any person with enough knowledge and will can find and exploit those bugs. Those with malicious intent can then go on to propagate that exploit mercilessly, as was seen with WannaCry, which can lead to millions of dollars in losses worldwide, as well as the shutdown of critical systems. Though exploits can be harmless, without publicly-available proof-of-concepts, or without self-propagation, they are still not worth taking the chance; a loss in system availability can be as critical as a loss of system integrity.

Works Referenced

Alam, D., Zaman, M., Farah, T., Rahman, R., & Hosain, M. S. (2017, July). Study of the dirty copy on write, a linux kernel memory allocation vulnerability. In *2017 International Conference on Consumer Electronics and Devices (ICCED)* (pp. 40-45). IEEE.

Chen, Q., & Bridges, R. A. (2017, December). Automated behavioral analysis of malware: A case study of wannacry ransomware. In *2017 16th IEEE International Conference on machine learning and applications (ICMLA)* (pp. 454-460). IEEE.

Demystifying BlueKeep and Remote Desktop Protocol (RDP) Vulnerability. (2020, January 17).

CERTStation Blog. https://certstation.com/blog/demystifying-bluekeep-remote-desktop-protocol-rdp-vulnerability/

"DirtyCOW Race Condition Attack." Programmer.help, 27 June 2020, programmer.help/blogs/dirty_cow_race_condition_attack.html. Accessed 8 Dec. 2023.

Donaldson, Samantha. "WannaCry Ransomware: Who It Affected and Why It Matters." Red Hat Developer, 19 May 2017, developers.redhat.com/blog/2017/05/19/wannacry-ransomware-who-it-affected-and-why-it-matters.

Greenberg, A. (2019, November 3). *The First BlueKeep Mass Hacking Is Finally Here—but Don't Panic*. Wired; WIRED. https://www.wired.com/story/bluekeep-hacking-cryptocurrency-mining/

Hare, Andrew. "What Is Copy-On-Write?" Stack Overflow, 10 Mar. 2009, stackoverflow.com/questions/628938/what-is-copy-on-write. Accessed 8 Dec. 2023.

Hsiao, S. C., & Kao, D. Y. (2018, February). The static analysis of WannaCry ransomware. In *2018 20th international conference on advanced communication technology (ICACT)* (pp. 153-158). IEEE.

Kumar, M. S., Ben-Othman, J., & Srinivasagan, K. G. (2018, June). An investigation on wannacry ransomware and its detection. In *2018 IEEE Symposium on Computers and Communications (ISCC)* (pp. 1-6). IEEE.

Martin, G., Ghafur, S., Kinross, J., Hankin, C., & Darzi, A. (2018). WannaCry—a year on. *Bmj*, *361*.

Microsoft. "Microsoft Security Bulletin MS17-010 - Critical." Learn.microsoft.com, 14 Mar. 2017, learn.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010.

Mohurle, S., & Patil, M. (2017). A brief study of wannacry threat: Ransomware attack 2017. *International journal of advanced research in computer science*, *8*(5), 1938-1940.

Pearson, Thaddeus. "Dirty Cow." Www.cs.toronto.edu, 5 Nov. 2021, www.cs.toronto.edu/~arnold/427/18s/427_18S/indepth/dirty-cow/index.html. Accessed 7 Dec. 2023.

QuinnRadich. (2019, August 23). Remote Desktop Services virtual channels - Win32 apps. Learn.microsoft.com. https://learn.microsoft.com/en-us/windows/win32/termserv/terminal-services-virtual-channels

SentinelOne. "Eternalblue | the NSA-Developed Exploit That Just Won't Die." SentinelOne, 27 May 2019, www.sentinelone.com/blog/eternalblue-nsa-developed-exploit-just-wont-die/.

Sheldon, Robert, and Jessica Scarpati. "What Is the Server Message Block (SMB) Protocol? How Does It Work?" SearchNetworking, Aug. 2021, www.techtarget.com/searchnetworking/definition/Server-Message-Block-Protocol.

"The WannaCry Encryption Malware Attack." Natural Networks, Inc., 23 May 2017, www.naturalnetworks.com/wannacry-encryption-malware-attack/. Accessed 7 Dec. 2023.

Van Impe, K. (2019, June 14). *How to Patch BlueKeep and Get to Know Your Company's Critical Assets*. Security Intelligence. https://securityintelligence.com/articles/how-to-patch-bluekeep-and-get-to-know-

your-companys-critical-assets/
What was the WannaCry ransomware attack?. CloudFlare. (n.d.). https://www.cloudflare.com/learning/security/ransomware/wannacry-ransomware/
Wilson, Jake, and Nimesha Jayawardena. “Dirty Cow.” Www.cs.toronto.edu, www.cs.toronto.edu/~arnold/427/18s/427_18S/indepth/dirty-cow/index.html. Accessed 7 Dec. 2023.