PROJECT I

Operating System Vulnerabilities and Bugs

## Abstract

This paper is an overview of operating system vulnerabilities and bugs with introduction of Common Vulnerabilities and Exposures glossary (CVE), six vulnerability types, vulnerabilities analyzing, and bugs in operating system lifetime.

## Introduction

When we say operating system, its vulnerability is always needed to be considered. Since the kernel is fully trusted, the publicly released software is a moderate source of malware. So, the Common Vulnerabilities and Exposures glossary (CVE), a project focused on storing information about security vulnerabilities and exposures, cataloging them according to distinctive identifiers and providing them with unique IDs [1], is introduced in this article.

CVE is a dictionary standardized by which totally different security advisories, bug trackers and databases. These resources obtained a unique baseline in same vulnerabilities and same format, and keep updating and communicating over time. Once documented, a unique ID will be provided by MITRE Corporation, a non-profitable company that works across government to tackle with safety problems. Not only MITRE can issue CVE numbers, many commercial entities (about 98[[1]](#footnote-1)) act as CVE Numbering Authorities (CNA), including CERT Coordination Center [1]. CVE full numbering rule, according to Anat Richter, is: “CVE + Year + 4digit serial number = CVE-2018-8589” [1, p. 1].

Once CVE is reported by individual developers to any of CNAs, they need to know the vulnerabilities’ security severity. So Common Vulnerability Scoring System, also referred to as CVSS score, is attached with every CVE. Its main purpose is to establish standards for measuring the severity of vulnerabilities and determine the priority of handling them. Thus, CVSS is effective in quickly performing vulnerability prioritization and screening vulnerabilities [2].

### Vulnerability Type

There are several vulnerabilities by type:

**Code Execution**: Some software systems allow to upload file extensions to a web server. This results in code execution if a file is executable [3].

**Overflow**: The overflow is generally referred to as stack overflow, which occurs when a function copies data into a buffer without checking the bound [4].

**Memory Corruption**: A memory corruption attack such as, buffer overflow, heap corruption, and format string, and once the attacker finds out the exploit, they can make program to crash [5].

**Bypass Something**: Bypassing something such as authentication checks in Linux, and allows an attacker to gain the same privileges as a legitimate user of a system [6].

**Gain Information**: The system configuration or software vulnerability is exploited to an expose system or network access information, which may lead to a successful attack [7].

**Gain Privileges**: The attackers successfully gain kernel-level privileges without having to do authentication check [8].

### Vulnerability Analysis

About the [9], the approach author uses to communicate with readers about macOS/iOS Vulnerability (CVE-2019-6231) [9] is that:

1. First and foremost, the author states the CVE details in general, report the vulnerability and give out a “quick look” [9, p. 1] of where was the “hole” located.
2. Then he gives out the crash log of one of the process and show the thread name.
3. Show the thread implemented in code and explain the mechanism of invoked functions (Analysis of where the vulnerability is originated).
4. Show the “Proof of Concept” (in C code) [9, p. 2].
5. Show the “Root of Cause” (there is an array index overflow) [1, p.3], and analysis how to solve them.
6. Gives out a general solution of this vulnerability, and compare the code “before patch and after patch” [9, p. 3].
7. Finally, gives out conclusion, affected version and references [9. p. 4].

And as for [10], the approach that author uses to communicate with readers about WinRAR Vulnerability (CVE-2018-20250) [10] is that:

1. Firstly, state the CVE details, just as what the author did in [9] in the first part, report it and summaries the vulnerability and its causes and consequences.
2. A very important component of the attack is explained and mentioned several times through out the whole blog: the ACE files [10].
3. Illustrate the “attack chain” [10, p. 1] and give flow chart to show in details.
4. Analysis the attack by every step the user may lure to do and show what is running under the hood, attached with screen shot in details.
5. Because the malicious macro runs based on PowerShell a lot [10], so after analysis mechanism of the ACE file attack, the author further interpret PowerShell and how does the vulnerability “help” the attacker to get system compromised without checking privileges.
6. Introduce the newly updated and integrated protection to solve the exploit and its new features.
7. Finally, gives out affected files and URLs (indicators of compromise) [10].

Those analysis of vulnerabilities represent a two groups of people who report CVE problems: The first part [9] is suited for individuals to solve the simple vulnerability exploits, and the second part [10] is suited for a development security group to come up with an integrated solution of a critical vulnerability.

### Operating System Bugs Discussion

## References

[1] Anat Richter, “What Is a CVE Vulnerability and How To Understand Its Details,” Blog, 7 Jan. 2019;

resources.whitesourcesoftware.com/blog-whitesource/what-is-cve-vulnerability

[2] Forum of Incident Response and Security Teams (FIRST), “Common Vulnerability Scoring System SIG,” Website, May 2019; www.first.org/cvss.

[3] Christey, S. and Martin, R. A. 2007. “Vulnerability Type Distributions in CVE,” Mitre Report, May 2007, pp. 138.

[4] Christiansen, M. 2007. “Stack Based Overflows: Detect & Exploit,” SANS Institute.

[5] Chen, S., Xu, J., Nakka, N., Kalbarczyk, Z. Iyer, R. K. 2005. “Defeating Memory Corruption Attacks via Pointer Taintedness Detection,” International Conference on Dependable Systems and Networks, Yokohama, Japan, pp. 378-387.

[6] Tajpour, A., Ibrahim, S., Sharifi, M. 2012. “Web Application Security by SQL Injection Detectiontools,” IJCSI International Journal of Computer Science Issues (9:2), pp. 332–339.

[7] Alan Litchfield and Abid Shazad, “A Systematic Review of Vulnerabilities in Hypervisors and Their Detection” in *Twenty-third Americas Conference on Information Systems*, Boston, 2017, p. 1.

[8] Provos, N. Friedl, M. Honeyman, P. 2003. “Preventing Privilege Escalation,” USENIX Security (3).

[9] Kai Lu, “Detailed Analysis of macOS/iOS Vulnerability CVE-2019-6231,” blog, 24 Jan. 2019; www.fortinet.com/blog/threat-research/detailed-analysis-of-macos-ios-vulnerability-cve-2019-6231.html.

[10] Rex Plantado, “Analysis of a targeted attack exploiting the WinRAR CVE-2018-20250 vulnerability,” blog, 10 Apr. 2019; www.microsoft.com/security/blog/2019/04/10/analysis-of-a-targeted-attack-exploiting-the-winrar-cve-2018-20250-vulnerability.

1. MITRE Coporation, “CVE-CVE Numbering Authorities,” May 2019; cve.mitre.org/cve/cna.html#submitting\_cve\_entry\_info [↑](#footnote-ref-1)