**CVE-2021-3156**

**Sudo Vulnerability**

Author: S.A.W.N. Subasinghe

Supervisor: Dr. Lakmal Rupasinghe

Reg. No.: **MS19814384**

Offensive Hacking: Tactical & Strategic

IE6052

Sri Lanka Institute of Information Technology

M.Sc. in IT



Contents

[CVE-2021-3156 - Sudo Vulnerability 2](#_Toc72014137)

[Introduction 2](#_Toc72014138)

[CVE-2021-3156 2](#_Toc72014139)

[Vulnerability Discovery 3](#_Toc72014140)

[Fuzzing 3](#_Toc72014141)

[Challenges 3](#_Toc72014142)

[Exploitation 6](#_Toc72014143)

[Mitigation 6](#_Toc72014144)

[Conclusion 7](#_Toc72014145)

[References 7](#_Toc72014146)

# CVE-2021-3156 - Sudo Vulnerability

## 

## Introduction

In present, computer security is a big area to be concerned with the rapid increment of cybercrimes and malicious activities. Most of the cybercrimes are happened due to the security loopholes in the system security which is technically called system vulnerabilities which can be define as a weakness in the system, which an malicious intruder exploits a computer system to compromise the security, or gain control a system without proper authorized privileges.

As above mentioned the intruders gain access to a vulnerable systems by leveraging security gaps, which are caused by errors or bugs in the operating systems or system's functionality. Also security applications cannot operate effectively as it is when the operating system is vulnerable. [1]

This document concludes the linux based vulnerability which related to the sudo command.

# CVE-2021-3156

Basically, there are two main method to launch applications with administrative privileges that’s depend on the linux distributions. This can be done though switching to the super user (root) with the su command, or sudo command. Some distributions such as Fedora, Red Hat, openSuSE are enabled the root user by default.

The sudo users who required to authenticate must be included in the sudoers file located in /etc. To modify the sudoers file users can be used “visudo” command, when using the sudo command users must authenticate themselves with particular user’s password and not required to use the root password itself. [2]

**The CVE-2021-3156 vulnerability is related with the sudo.**

The version released prior to 1.9.5p2 sudo has an off-by-one vulnerability that can cause a heap-based buffer overflow, this is enabling privilege escalation to root via "sudoedit -s" and a command-line argument that ends with a "\"character.

The way sudo parses command line arguments has a heap-based buffer overflow and non-privilege user are allowed to execute the sudo commands bypassing the authentication, therefore this is categorized as a privilege escalation. [3]

This vulnerability poses the greatest risk to data security and integrity, as well as availability of the system.

# Vulnerability Discovery

## Fuzzing

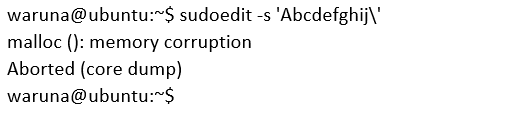
A method of analysis for applications in which provide the target program with arbitrary, invalid data. The submission is then focused mainly on for any possible errors. Memory leakage, crashing, and other unusual behaviors could occur for previously unknown niche test cases that are difficult to observe using manual testing. In here researcher used to fuzzing technique to discover the vulnerability with American Fuzzy Lop. [4]

There are few changes can be identify when fuzzing with AFI

## Challenges

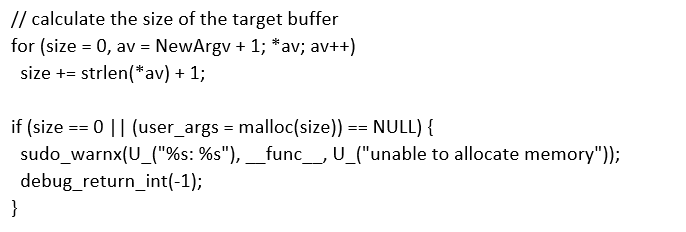
* Program arguments cannot be fuzzed by afl. The intended binary must be instrumented, such as with the experimental argv-fuzz-inl.h library.
* sudo command has non-identical behavior when initiated with sudoedit command. One has to patch the progname.c utility sine argv[0] not inuse.
* Due to the various functionality of the sudo command when executed as root or an unpriviledged user fuzzing set up should be customized getuid to 1000 [5]

https://medium.com/@ayushpriya10/fuzzing-applications-with-american-fuzzy-lop-afl-54facc65d102

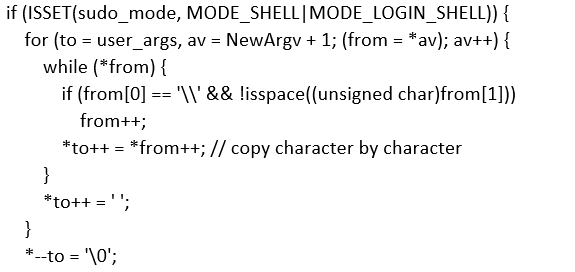


**Vulnerable “**[**set\_cmnd()**](https://github.com/sudo-project/sudo/blob/SUDO_1_8_31p1/plugins/sudoers/sudoers.c#L852)**“ function**

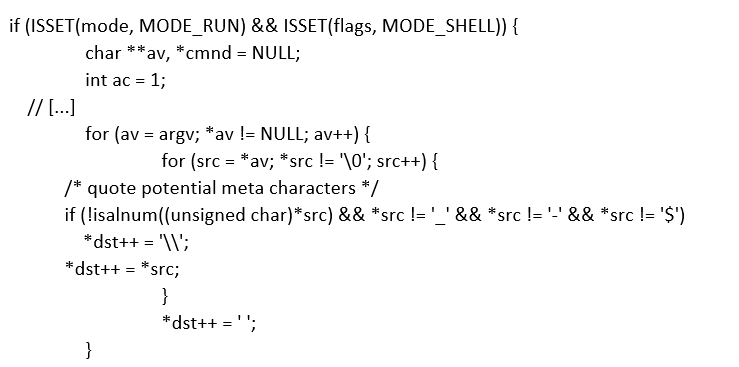
Initially, in the NewArgv array, loop continue through all the strings and sums up the length, which results in the allocation of a target buffer of that size.



After above section it initiates to copy characters one by one loop into the target buffer (user\_args).If it is included a “\”, then it will skip and copy the next character and continue.Also there is a possibility of copying a string out of bounds, when a “\” is typing before the terminating null-byte. Example string: Abcdefghij\



Even though the code mentioned above is appear as not well protected. In *parse\_args.c* data is pass through an escape loop and add additional “\”. (This will add additional “\” and final output modify as 'Abcdefghij\\’ then set\_cmnd() function will work safely as usual.



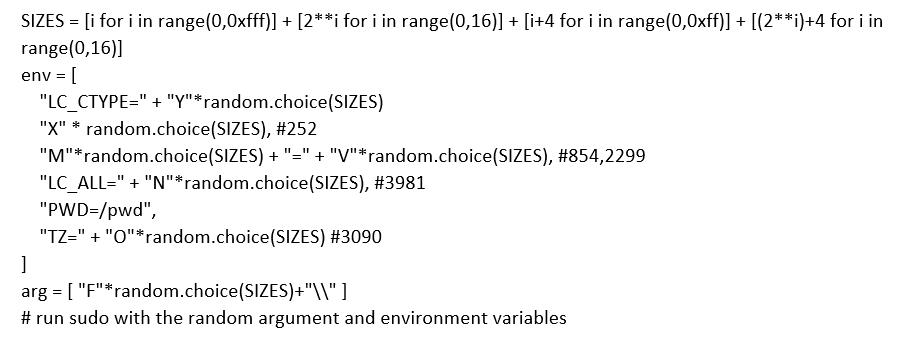
When it analyzing more deeply at the conditions when either loop is invoked. Sudo has many different modes which can set though the arguments. It can be easily identified when considering the if-conditions preceding the two loops.

Here the challenge is finding the way to put sudo mode which can runs the set\_cmnd() loop which is not initially starting though the escape loop in “parse\_args” but this is possible with by using sudoedit –s command. This clearly indicates a buffer overflow which open access to overwrite any object coming after the vulnerable user\_args in system memory locations.

To point the specific overwritten code for an exploit, initially it should be analyzed by correctly identifying the user controls which is effected heap allocations. After that a script can be developed to

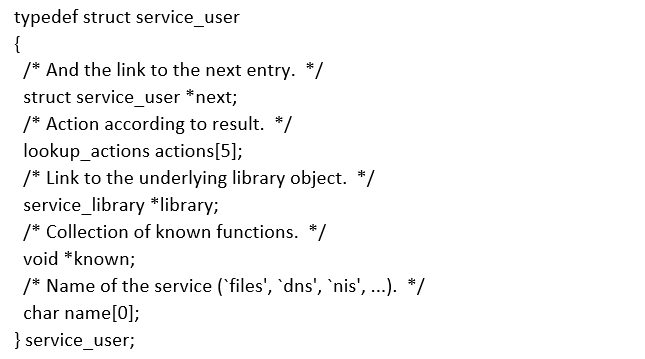
Brute-force and fuzzes to find the various sudo crashing heap layouts

It's easy to fuzz a lot of different inputs and gather backtraces where they fail using a script.



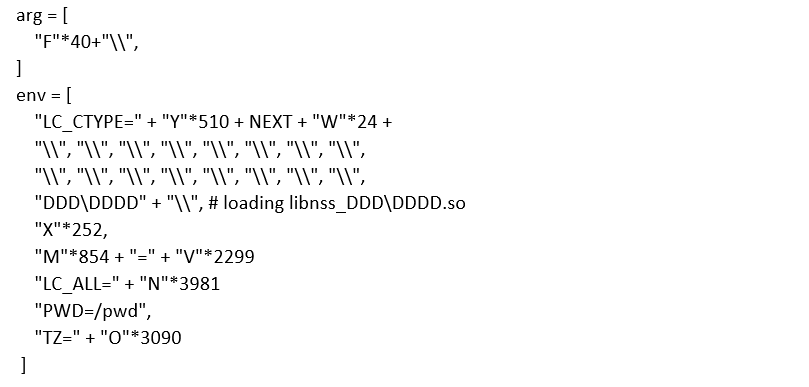
There is a particular function {In this Eg. nss\_lookup\_function() } can be identified when going through the crash locations. Using nss\_lookup\_function()might be able to overflow a value that controls what function is looked up and executed.

The service\_user object inside the nss\_lookup\_function()will ten call dlopen that loads an external library. Thus, Overflowing the name of the object will control the name of the loaded shared library is determined by the overflowing the name of the object [6]

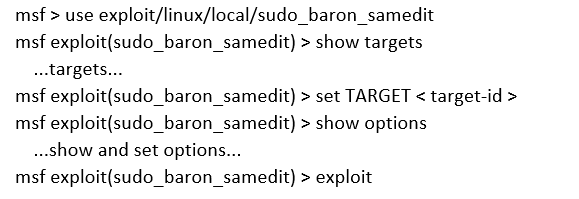


## Exploitation

The perfect condition that creates the ideal heap layout can be identified by analyzing the fuzzing inputs and backtraces,



Exploit the Sudo Heap-Based Buffer Overflow vulnerability using MetasploitModule [7]



## Mitigation

To mitigate the vulnerability all the product already release the security/kernel patch updates.

Therefore it is highly recommended to update all the products to the latest version to overcome the security issues related to the old legacy versions.

# Conclusion

This heap-based buffer overflow vulnerability was identified after almost ten years later. Therefore, it is highly recommended to check the all the applications and operating system manually for the security issues regularly, apart from traditional vulnerability assessment and pen testing. And also security patches should be updated and systems must be restarted if necessary.

Another most important thing is all the servers must have additional layer of security to mitigate the risk, such as implementing the latest, up to date and properly fine-tuned perimeter layer security devices and main firewalls to minimize the exposure to public as well as internal threads. Therefore to protect against the cyber threat it required great involvement of several IT teams.

# References

|  |  |
| --- | --- |
| [1] | "IEEE Long Island Systems, Applications and Technology Conference, LISAT," *Linux kernel OS local root exploit,* 2017. |
| [2] | "CVE-2021-3156," The MITRE Corporation, 2021. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-3156. [Accessed 11 05 2021]. |
| [3] | "Metasploit exploit/linux/local/sudo\_baron\_samedit," InfosecMatter , 2021. [Online]. Available: https://www.infosecmatter.com/metasploit-module-library/?mm=exploit/linux/local/sudo\_baron\_samedit. [Accessed 12 05 2021]. |
| [4] | "Fuzzing Applications with American Fuzzy Lop (AFL)," 2021. [Online]. Available: https://medium.com/@ayushpriya10/fuzzing-applications-with-american-fuzzy-lop-afl-54facc65d102. [Accessed 12 05 2021]. |
| [5] | "sudo - Baron Samedit CVE-2021-3156," liveoverflow, [Online]. Available: https://liveoverflow.com/critical-sudo-vulnerability-walkthrough-cve-2021-3156/. [Accessed 11 05 2021]. |
| [6] | "Baron Samedit: Heap-based buffer overflow in Sudo (CVE-2021-3156)," Qualys Security Advisory, 2021. [Online]. Available: https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-sudo.txt. [Accessed 10 05 2021]. |
| [7] | "Sudo Heap-Based Buffer Overflow," rapid7, 2021. [Online]. Available: https://www.rapid7.com/db/modules/exploit/linux/local/sudo\_baron\_samedit/#:~:text=A%20heap%20based%20buffer%20overflow,2011%20and%20affects%20version%201.8. [Accessed 14 05 2021]. |
| [8] | "2017 IEEE Long Island Systems, Applications and Technology Conference, LISAT 2017," 2017. |