

## How to Limit Display of Other User's Processes in Solaris 10

by: Glenn M. Brunette, Jr., 10/13/2004

<http://www.securitydocs.com/library/2640>

This entry is a continuation of my list of lesser known and/or publicized security enhancements to the Solaris 10 OS. In this update, I will be talking about how to restrict the output of the [ps\(1\)](#) command such that users can only see processes that they own. This is a very useful capability especially for ISPs and other organizations that do not want to allow users to see what other users are doing.

This new feature would not have been possible without the introduction of [process privileges](#) into the Solaris 10 OS. For a great overview of process privileges, see [Casper Dik's blog entry](#) on this topic. Be sure to read his article to get a more in depth understanding of process privileges.

So, before we begin, let's see what the output of `ps -aef` might look like for an average user (in this case, *gmb*):

```
$ ps -aef
  UID   PID  PPID  C   STIME TTY          TIME CMD
  root     0     0   0   Sep 23 ?           0:07 sched
  root     1     0   0   Sep 23 ?           0:01 /sbin/init
  root     2     0   0   Sep 23 ?           0:00 pageout
  root     3     0   0   Sep 23 ?           2:31 fsflush
  root   393     1   0   Sep 23 ?           0:00 /usr/sbin/auditd
  root     7     1   0   Sep 23 ?           0:11 /lib/svc/bin/svc.startd
  root     9     1   0   Sep 23 ?           0:19 svc.configd
  root   176     1   0   Sep 23 ?           0:00 /usr/sbin/syslogd
  root    64     1   0   Sep 23 ?           0:00 /usr/sbin/nscd
daemon   91     1   0   Sep 23 ?           0:02 kcfld
  root   170     1   0   Sep 23 ?           0:01 /usr/lib/utmpd
  gmb  1795  1792   0 22:17:26 pts/1       0:00 -sh
  root  1527     7   0 00:53:24 console    0:00 /usr/bin/login
  root    82     1   0   Sep 23 ?           0:00 /usr/lib/sysevent/syseventd
smmsp   334     1   0   Sep 23 ?           0:00 /usr/lib/sendmail -Ac -q15m
daemon  137     1   0   Sep 23 ?           0:06 /usr/sbin/rpcbind
  root    84     1   0   Sep 23 ?           0:00 /usr/lib/picl/picld
  root  1601  1527   0 07:36:19 console    0:00 -sh
  root   181     1   0   Sep 23 ?           0:04 /usr/lib/inet/inetd start
  root   281     1   0   Sep 23 ?           0:00 /usr/lib/nfs/mountd
  root   187     1   0   Sep 23 ?           0:00 /usr/sbin/cron
  root  1402     1   0 00:26:14 ?           0:00 /usr/lib/ssh/sshd
daemon  289     1   0   Sep 23 ?           0:00 /usr/lib/nfs/nfsd
daemon  264     1   0   Sep 23 ?           0:00 /usr/lib/nfs/statd
  root   303     1   0   Sep 23 ?           0:00 /usr/lib/fm/fmd/fmd
daemon  268     1   0   Sep 23 ?           0:00 /usr/lib/nfs/lockd
  root   291     1   0   Sep 23 ?           0:00 /usr/lib/autofs/automountd
  gmb  1799  1795   0 22:17:28 pts/1       0:00 ps -aef
  root  1789   181   1 22:17:19 ?           0:00 /usr/sbin/in.telnetd
  root  1792  1789   1 22:17:19 pts/1       0:00 login -p -h 10.1.1.100 -d /dev/pt
  root   335     1   0   Sep 23 ?           0:06 /usr/lib/sendmail -bd -q15m
daemon  296     1   0   Sep 23 ?           0:00 /usr/lib/nfs/nfsmapid
```

As you can see, the *gmb* user can see not only his processes but also those of the *root*, *daemon*, and *smmsp* accounts. We can change this behavior either globally or on a per-user basis. Just as we

discussed in the [Solaris 10 Account Lockout](#) entry, we can use user-specific changes to force a subset of users to comply with some policy or use the user-specific changes to make exceptions for those users. The flexibility of this format allows it to be adapted quite easily to the needs of many organizations.

For our first example, we will illustrate how the global change can be made. So do this, we must edit the [/etc/security/policy.conf](#) file, uncomment the PRIV\_DEFAULT entry and set its value as follows:

```
PRIV_DEFAULT=basic,!proc_info
```

For those not familiar with the *proc\_info* privilege, you can find more information about it using the [ppriv\(1\)](#) command:

```
# ppriv -l -v proc_info
proc_info
    Allows a process to examine the status of processes other
    than those it can send signals to. Processes which cannot
    be examined cannot be seen in /proc and appear not to exist.
```

This is all that you need to do to globally configure your Solaris 10 system so that users will only be able to see processes that they own. (Note that this will obviously not apply to *root* who by default has all privileges or to other users or processes that have been explicitly given the *proc\_info* privilege.) Regardless, it is still a very quick and effective way to limit what processes users may see.

Returning to the *gmb* account example, I re-login to the system and again run the *ps -aef* command, but this time I receive different results:

```
$ ssh -l gmb sampleHost
gmb@sampleHost's password:
Last login: Fri Sep 24 22:25:18 2004 from 10.1.1.100
Sun Microsystems Inc. SunOS 5.10 s10_67 May 2004
$ ps -aef
    UID  PID  PPID  C   STIME TTY          TIME CMD
    gmb  1823  1819  0   22:30:17 pts/1        0:00 ps -aef
    gmb  1819  1815  0   22:30:14 pts/1        0:00 -sh
$
```

As you can see, the *gmb* user may now only see his own processes. Way cool. Next, to illustrate the per-user configuration ability, I will leave this global configuration in place and use the per-user configuration ability to allow the *gmb* user to view processes owned by other users. This is just an example of how exceptions can be implemented. The same process could be used to enable this feature for just a user or subset of users on the system.

To accomplish this task, we go back to the [user\\_attr\(4\)](#) file. In this file, we will modify the existing entry for the *gmb* user (or create one if one had not already been there). The following example illustrates the change that needs to be made. Specifically you need to add the *defaultpriv* entry to specify the default list of privileges that will be available to this user. By modifying this parameter, you will change the default set of privileges available to a user (by either adding or removing privileges as needed.) In this case, we are returning the user's default set of privileges to *basic* from *basic,!proc\_info*.

```
gmb::::lock_after_retries=no;defaultpriv=basic
```

So, let's see if this really works. In the following example, we will confirm the configuration of the system, login to the system as the *gmb* user, and run the *ps -aef* command to verify that the *gmb* user can see processes owned by other users.

```
# grep "^PRIV_DEFAULT=" /etc/security/policy.conf
PRIV_DEFAULT=basic,!proc_info
# grep "^gmb:" /etc/user_attr
gmb::::lock_after_retries=no;defaultpriv=basic
# ssh -l gmb localhost
Password:
Last login: Fri Sep 24 22:37:55 2004 from 10.1.1.100
Sun Microsystems Inc. SunOS 5.10 s10_67 May 2004
$ id -a
uid=100(gmb) gid=1(other) groups=1(other)
$ ps -aef
  UID   PID  PPID  C   STIME TTY          TIME CMD
  root     0     0   0   Sep 23 ?           0:07 sched
  root     1     0   0   Sep 23 ?           0:01 /sbin/init
  root     2     0   0   Sep 23 ?           0:00 pageout
  root     3     0   0   Sep 23 ?           2:33 fsflush
  root   393     1   0   Sep 23 ?           0:00 /usr/sbin/auditd
  root     7     1   0   Sep 23 ?           0:11 /lib/svc/bin/svc.startd
  root     9     1   0   Sep 23 ?           0:19 svc.configd
  root   176     1   0   Sep 23 ?           0:00 /usr/sbin/syslogd
  root    64     1   0   Sep 23 ?           0:00 /usr/sbin/nscd
daemon   91     1   0   Sep 23 ?           0:02 kcfld
  root   170     1   0   Sep 23 ?           0:01 /usr/lib/utmpd
  root  1900  1402   7 22:42:05 ?           0:02 /usr/lib/ssh/sshd
  root  1527     7   0 00:53:24 console    0:00 /usr/bin/login
  root    82     1   0   Sep 23 ?           0:00 /usr/lib/sysevent/syseventd
smmsp    334     1   0   Sep 23 ?           0:00 /usr/lib/sendmail -Ac -q15m
daemon  137     1   0   Sep 23 ?           0:06 /usr/sbin/rpcbind
  root    84     1   0   Sep 23 ?           0:00 /usr/lib/picl/picld
  root  1601  1527   0 07:36:19 console    0:00 -sh
  root   181     1   0   Sep 23 ?           0:04 /usr/lib/inet/inetd start
  root   281     1   0   Sep 23 ?           0:00 /usr/lib/nfs/mountd
  root   187     1   0   Sep 23 ?           0:00 /usr/sbin/cron
  root  1402     1   0 00:26:14 ?           0:00 /usr/lib/ssh/sshd
daemon  289     1   0   Sep 23 ?           0:00 /usr/lib/nfs/nfsd
daemon  264     1   0   Sep 23 ?           0:00 /usr/lib/nfs/statd
  root   303     1   0   Sep 23 ?           0:00 /usr/lib/fm/fmd/fmd
daemon  268     1   0   Sep 23 ?           0:00 /usr/lib/nfs/lockd
  root   291     1   0   Sep 23 ?           0:00 /usr/lib/autofs/automountd
  gmb  1912  1908   0 22:42:12 pts/1      0:00 ps -aef
  root   335     1   0   Sep 23 ?           0:06 /usr/lib/sendmail -bd -q15m
daemon  296     1   0   Sep 23 ?           0:00 /usr/lib/nfs/nfsmapid
  gmb  1908  1900   0 22:42:10 pts/1      0:00 -sh
  root  1899  1601   6 22:42:05 console    0:02 ssh -l gmb localhost
```

It worked! That was pretty easy, right? This is just one very small example of how you can use process privileges in your daily lives. I will try to add more interesting examples of practical uses for process privileges in the future.

Before ending, I do want to highlight that while these examples focused on the *ps(1)* command - other process related commands will also be impacted such as *ptree(1)*, *pcrred(1)*, *pmap(1)*, *psig(1)*, etc.

Further, as a user running without the *proc\_info* privilege, you will not even be able to see other processes in the */proc* directory:

```
$ id -a
uid=101(foo) gid=1(other) groups=1(other)
$ ppriv $$
1915:  -sh
flags =
      E: basic,!proc_info
      I: basic,!proc_info
      P: basic,!proc_info
      L: all
$ ls -l /proc
total 4
dr-x--x--x  5 foo      other      832 Sep 24 22:52 1915
dr-x--x--x  5 foo      other      832 Sep 24 22:56 1932
```

I hope you enjoyed this article and please watch this space for new discussion of Solaris 10 security features.