

## Solaris 10 Account Lockout ("Three Strikes!")

by: Glenn M. Brunette, Jr., 10/11/2004

<http://www.securitydocs.com/library/2637>

The next item of my list of lesser known and/or publicized security enhancements to the Solaris 10 OS is account lockout. Account lockout is the ability of a system or service to administratively lock an account after that account has suffered "n" consecutive failed authentication attempts. Very often "n" is three hence the "three strikes" reference.

Recall from yesterday's entry on [non-login and locked accounts](#) that there is in fact a difference. Locked accounts are not able to access any system services whether interactively or through the use of delayed execution mechanisms such as cron(1M). So, when an account is locked out using this capability, only a system administrator is able to re-enable the account, using the [passwd\(1\)](#) command with the "-u" option.

Account lockout can be enabled in one of two ways. The first way will enable account lockout globally for all users. The second method will all more granular control of which users will or will not be subject to account lockout policy. **Note that the account lockout capability will only apply to accounts local to the system.** We will look at both in a little more detail below.

Before we look at how to enable or disable the account lockout policy, let's first take a look at how you configure the number of consecutive, failed authentication attempts that will serve as your line in the sand. Any number of consecutive, failed attempts beyond the number selected will result in the account being locked. This number is based on the *RETRIES* parameter in the [/etc/default/login](#) file. By default, this parameter is set to 5. You can certainly customize this parameter based on your local needs and policy. By default, the [Solaris Security Toolkit](#) will set the *RETRIES* parameter to 3.

Now that we know how to define how many consecutive, unsuccessful authentication attempts we will allow, let's take a look at how you can enable the account lockout policy globally. This policy can be altered using the *LOCK\_AFTER\_RETRIES* variable in the [/etc/security/policy.conf](#) file. Just as it sounds, if you set this parameter to *YES*, then the account lockout policy is enabled for all users on the system (unless there is a user override which we will talk about in a minute). By default, this parameter is set to *NO* which means that account lockout is not enabled. So, let's try a simple example. First, I created a test account called *gmb*. Next, I set the *LOCK\_AFTER\_RETRIES* parameter in [/etc/security/policy.conf](#) to *YES*. To see, how this feature works, I attempted to authenticate to a system (and failed) using three different services:(1) TELNET, (2) FTP and (3) RLOGIN. I failed the login attempt for each of these services (in turn) twice with the exception of RLOGIN since after the fifth failed attempt the account was locked. I ran this test from the system's console so that the log messages could be injected into the output stream to give you a better idea of what was happening. Here is the actual log of the test that was run:

```
# telnet localhost
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
login: gmb
Password:
Login incorrect
login: gmb
```

```

Password:
Login incorrect
login:
Connection to localhost closed by foreign host.
# ftp localhost
Connected to localhost.
220 sampleHost FTP server ready.
Name (localhost:root): gmb
331 Password required for gmb.
Password:
530 Login incorrect.
Login failed.
ftp> user gmb
331 Password required for gmb.
Password:
530 Login incorrect.
Login failed.
ftp> quit 221 Goodbye.
# rlogin -l gmb localhost Password:
Sep 23 23:23:47 sampleHost login: Excessive (5)

login failures for gmb: locking account. Login

incorrect
login:

```

As you can see, after the fifth attempt, the *gmb* account was locked. This can also be verified by looking at the `shadow(4)` file entry for that account:

```

# grep "^gmb:" /etc/shadow
gmb:*LK*R12OfCMPngtJQ:12685::::::5

```

You can see that the account has been locked and that a record of the number of failures is available in the last column. From the [shadow\(4\)](#) manual page, the last field (called "flag") stores the failed login count in the low order four bits while reserving the remainder for future use. This means that you can also look at individual `shadow(4)` entries and see how many consecutive failed authentication attempts have been made per user. For example, you could do the following to see how many users have had failed authentication attempts since their last successful login:

```

# awk -F: '$NF >= 1 { print; }' /etc/shadow
gmb:*LK*R12OfCMPngtJQ:12685::::::5
foo:02YZb5ZaMrck:12685::::::2
bar:XF0Ggjq1c6tYQ:12685::::::1
baz:.VxOG4ytNE8es:12685::::::3

```

If a user who has had failed authentication attempts is finally able to successfully login to the system, that user will be presented with a message like:

```

# telnet localhost
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
login: baz
Password:

```

```
Warning: 3 failed login attempts since last
successful login.
Last login: Thu Sep 23 23:36:44 from localhost
```

This warning message is available for interactive login services (not FTP) and is very helpful in providing warning to users who may not have been responsible for the failed authentication attempts. It is important that you educate your users to not simply ignore these messages as they could be a symptom of an ongoing attack on their account.

Also, note that once a user has successfully authenticated to a system, the failed login count is reset:

```
# grep "^baz" /etc/shadow
baz:.VxOG4ytNE8es:12685:::~:
```

Note that the use of alternate authentication mechanisms such as rhosts or Secure Shell public key authentication will not reset the failed login count even on successful login. Should an account be locked however (either administratively or through the account lockout facility), the account would no longer be accessible even when using these alternate authentication methods. For example:

```
# grep gmb /etc/shadow
gmb:*LK*R12OfCMPngtJQ:12685:::~:
# rsh -l gmb localhost /bin/finger
account expired
```

or for Secure Shell...

```
# ssh -l gmb -i /export/home/gmb/.ssh/id_dsa

localhost
Enter passphrase for key

'/export/home/gmb/.ssh/id_dsa':
Sep 24 00:34:59 sampleHost sshd[1504]: Failed

publickey for gmb from 127.0.0.1 port 32801 ssh2
Password:
```

The second way in which account lockout can be configured is per-user in the `/etc/user_attr` file. Each user listed in the `/etc/user_attr` file can have an attribute defined called `lock_after_retries`. For a description of the format of this file, see the [user\\_attr\(4\)](#) manual page. By default, this value is set to `no`.

To configure account lockout for a specific user, simply add the `lock_after_retries` attribute with a value of `yes`. For example, let's assume you have an entry for user `gmb`:

```
gmb::::type=normal;profiles=FOO Security
Management;roles=secadm
```

To enable account lockout, you simple change the above line to:

```
gmb:::type=normal;profiles=FOO Security
Management;roles=secadm;lock_after_retries=yes
```

Let's take another view on this. Let's assume that the account lockout policy has been enabled globally using the method described above. You can then configure some users to be immune to this policy using this user-specific override. For example, if the *LOCK\_AFTER\_RETRIES* parameter was set to *YES* in */etc/security/policy.conf*, but you did not want the policy to apply to the *gmb* account, then you only need to make sure that the */etc/user\_attr* file contains an entry for the *gmb* account that sets the *lock\_after\_retries* attribute to *no* as in:

```
gmb:::lock_after_retries=no
```

Here is an example of how this works. I will attempt to access the *gmb* account with an invalid password five times using TELNET. In contrast to the above example, the account should not be locked and no account locked message should be generated. First, let's confirm we have our system configured correctly for this test:

```
# grep "^gmb:" /etc/shadow
gmb:h8HsRoqrneloQ:12685::::::::::::
# grep "^gmb:" /etc/user_attr
gmb:::lock_after_retries=no
# grep "^LOCK_AFTER_RETRIES="
```

```
/etc/security/policy.conf
LOCK_AFTER_RETRIES=YES
```

Now, let's see if the account actually gets locked after 5 failed authentication attempts.

```
# telnet localhost
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
login: gmb
Password:
Login incorrect
login: gmb
Password:
Login incorrect
login: gmb
Password:
Login incorrect
login: gmb
Password:
Login incorrect
login: gmb
Password:
Login incorrect
login: gmb
Password:
Login incorrect
Sep 23 23:51:46 sampleHost login: REPEATED LOGIN

FAILURES ON /dev/pts/1 FROM localhost, gmb
Connection to localhost closed by foreign host.
```

```
# grep "^gmb:" /etc/shadow
gmb:h8HsRoqrneloQ:12685::::::
```

Just as expected, the *gmb* account is immune from the account lockout policy that applies to other users on the system. This is in fact what is implemented by default for the *root* account. That is, even if account lockout is enabled globally (which is not the default), the *root* account is still immune from being locked out. This is done to prevent a malicious user from locking the *root* account out of the system. If you would like this policy to apply to the *root* account, then simply change the value of the *lock\_after\_retries* parameter to *yes* in the */etc/user\_attr* file.

This concludes another installment. As always, I hope you find this information useful in understanding how some of the new Solaris 10 security enhancements work and how they can be applied to solve real-world problems in your environment.