WORKING WITH PAM

CONNECTING TO THE LAB EQUIPMENT

Your instructor will give instructions on how to connect to your lab equipment.

EXERCISE 1 – CONFIGURING PASSWORD REQUIREMENTS

Perform the following steps on your virtual machine:

1. Login as **root**, with the password "**redhat**"
2. Configure the pwquality.so call in both /etc/pam.d/system-auth and /etc/pam.d/password-auth such that passwords are required to be a minimum of 12 characters in length and are required to have one 'other' character and one 'digit

```
[root@serverX ~] # vi /etc/pam.d/system-auth
...
password   requisite   pam_pwquality.so try_first_pass
local_users_only retry=3 authtok_type= ocredit=-1 dcredit=-1 minlen=12
```

3. ssh to localhost as **student**, login and change **student**'s password to something that will work, **i<3mylittleponies**.

```
[root@serverX ~]# ssh student@localhost
student@localhost's password: student
[student@serverX ~]$ passwd
Changing password for user student.
Changing password for student.
(current) UNIX password: student
New password: i<3mylittleponies
Retype new password: i<3mylittleponies
passwd: all authentication tokens updated successfully.
```

4. In the same open student user session, attempt to change the password to something that will not work, **L1b3r8yourself**.

```
[student@serverX ~]$ passwd
Changing password for user student.
Changing password for student.
(current) UNIX password: i<3mylittleponies
```

```
New password: L1b3r8yourself
BAD PASSWORD: The password contains less than 1 non-alphanumeric
characters
```

EXERCISE 2 – LOCKING ACCOUNTS AFTER FAILED LOGINS

Perform the following steps on your machine

1. Log into your machine as root.
2. Edit both the **/etc/pam.d/system-auth** and **/etc/pam.d/password-auth** files. Add a new rule calling **pam_tally2.so** which locks accounts, even root, after 3 failed login attempts. Users should be automatically unlocked after 3 minutes.

```
[root@serverX ~]# vi /etc/pam.d/system-auth
...
auth  required pam_tally2.so deny=3 even_deny_root unlock_time=180
...
```

   It is important that the **pam_tally2.so** line is inserted after the calls to **pam_env.so**, but before the lines calling to the authentication methods.

3. Use an ssh connection as **student** to localhost. Purposefully provide three erroneous passwords.

```
[root@serverX ~]# ssh student@localhost
student@localhost's password: WRONG
Permission denied, please try again.
student@localhost's password: WRONG
Permission denied, please try again.
student@localhost's password: WRONG
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
```

4. With your **pam_tally.so** rule triggered, verify the failed login count using the **pam_tally2** command.

```
[root@serverX ~]# pam_tally2
Login           Failures  Latest  failure     From
student             3    04/16/14 21:47:15  localhost
```

5. The **student** user account should now be locked for three minutes. Try to ssh to localhost as **student**, and provide the correct password to verify that **pam_tally2.so** is honoring the lock-out.

```
[root@serverX ~]# ssh student@localhost
student@localhost's password: i<3mylittleponies
Permission denied, please try again.
```

6. Wait for the three minute timeout, and again attempt to log in as student, again using the correct password.

```
[root@serverX ~]# ssh student@localhost
```

```
student@localhost's password: i<3mylittleponies
[student@server0 ~]$
```