SELinux Policy Editor RBAC(Role Based Access Control) guide (for Ver 2.0))

Yuichi Nakamura [*]

July 3, 2006

Contents

[*] himainu-ynakam@miomio.jp

This document describes how to use RBAC(Role Based Access Control) in seedit.

1 What is RBAC

1.1 Overview

SPDL supports configuration of RBAC(Role Based Access Control). In default policy, the domain for login user is unconfined_t. So, the behavior of user is not confined by SELinux.
To increase security of login user, RBAC is useful. By using RBAC, you can restrict behavior of users by assigning role to user. For example, you can assign role webmaster_r to user webmaster, and give him rights to do web master work.

1.2 How RBAC works in SELinux

How RBAC works is composed of 2 parts, one is assign role to user, second is assign domain to user shell.

(1) Assign role
When user logs in from login programs(login, sshd, gdm), login programs assign role to users. The rule that describes what kind of roles the user is allowed to use, is described in policy. Login programs assign roles referring to policy.For example, if user webmaster is allowed to use webmaster_r role, login program assign webmaster_r to user webmaster.

(2) Assign domain
Role is only strings, to confine behavior of user, domain must be given to user shell.
When user shell is launched domain is given according to role. For example, when user webmaster login as webmaster_r role, webmaster_t domain is given to user shell. webmaster_t domain is configured to be allowed homepage admin works.

2 Enable RBAC

To enable RBAC, use following command.

```
# seedit-rbac on
(It takes some minutes)
# reboot
```

By above commands, files necessary to configure RBAC is moved from /etc/seedit/policy/extra to /etc/seedit/policy, and seedit-load is run.Reboot is necessary because some domains become invalid, to fix this you have to reboot.

To disable RBAC, use following.

```
# seedit-rbac off
# reboot
```

3 Default RBAC Configuration

Following 3 roles are defined by default.

- sysadm_r
  It is role for administrator. It can work as unconfined domain sysadm_t. By default, only root can login as the role.
- staff_r
  It is role, to do not administrative work for administrative user. By default, only root can login as the role.
- user_r
  It is a role for normal users. By default, user_u can login as root. user_u is a user that is not configured RBAC, by default users except root.
- Attention

  (1) su command can not be used, except sysadm_r

  (2) Only user who can use sysadm_r can login from gdm.
  You need to add a lots of configurations to login from other roles, and it may decrease security. By default only root user can login from X. If other users want to login from X, you have to allow to use sysadm_r, but be careful, because behavior of such users are not confined by SELinux.

  (3) Configure to use sysadm_r
  Usually, you will not login root user directly. You will use su, you have to allow some user to use sysadm_r. For example to allow user ynakam to use sysadm_r, add following after *user root;* in /etc/seedit/policy/sysadm_r.sp.

  ```
  user ynakam;
  ```

4 Login by RBAC

4.1 Check role

When you enable RBAC, role are given to login user. You can see it by id command.

When you login as root user, staff_r role is given. root is allowed to use staff_r and sysadm_r, but login program give staff_r role.
Let's see it by id command.

```
# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),
3(sys),4(adm),6(disk),10(wheel)
context=root:staff_r:staff_t
```

```
context=root:staff_r:staff_t
```

shows role, staff_r is role. User shell is given domain according to role, in this case staff_t. Inside SELinux, domain is used for access control.
staff_r is given little access rights. You can not do any administration work in this role.
For example, you can not access homepage.

```
# cat /var/www/html/index.html
Permission denied
```

4.2 Change role

To do administrative work, you have to switch role to sysadm_r role. You can do it by newrole command, like below.

```
# newrole -r sysadm_r
Authenticating root
Password:
```

You have to enter password of current user(this case root). Then check role by id command.

```
# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),
3(sys),4(adm),6(disk),10(wheel)
context=root:sysadm_r:sysadm_t
```

Role is sysadm_r. Domain of user shell is sysadm_t. sysadm_t is unconfined domain, so you can do any work.
To switch role, the user must be allowed to use the role, if the user is not allowed to use the role, newrole will fail.
To allow user to use role, see next section.

5 Configuration elements for RBAC

To configure RBAC, it is better to understand how to describe.

5.1 role and user statements

To configure RBAC, there are 2 SPDL configuration elements, *role* and *user*.

- role *name of role*
  This means configurations is going to be done for specified role. Access right is given to corresponding domain. And the configuration filename must be *role name*.sp.

- user *user name*
  This allows user to use role. user_u user name is special, it is the default role of users that is not configured to use domain.How to allow to use sysadm_r, see section 5.3.

5.1.1 Example

Let's see example.

```
1:{
2:role webmaster_r;
3:user webmaster;
4:allow /var/www/** r,w,s;
}
```

Line 2 means, configurations between {} is for webmaster_r role.
Line 3 means, user name webmaster can use webmaster_r role.
Following, access rights are given to domain webmaster_t domain(Remember that webmaster_r role behaves as webmaster_t domain in SELinux system). Line 4 means, webmaster_t domain(This equals user that logined as webmaster_r role)is allowed to read, write under /var/www.

5.1.2 user_u user name

Let's see example of user_u user name.

```
Assume all configurations for RBAC are following.
* In sysadm_r.sp
{
role sysadm_r;
user root;
..
}
* In webmaster_r.sp
{
role webmaster_r;
user webmaster
..
}
* In user_r.sp
{
role user_r;
user user_u;
..
}
```

In above, 3 roles are configured. You can see, user root and webmaster are assigned role. In this case user_u is *all users except root and webmaster.*

5.2 Home directories

To configure access control to home directories, we could use ˜/ . In normal domain, it means all users home directories. But, for configuration of role, the meaning is different.
The rule is simple:

```
~/ means home directories for users that can use role
```

Let's see example.

```
{
1:role webmaster_r;
2:user web1;
3:user web2;
4:allow ~/** r,w,s;
```

In this case, line 4 is allow to write home directories for user1 and user2. So, when user web1/web2 login as webmaster_r role, they can read write their home directories, but can not access other users home directories.

```
{
1:role user_r;
2:user user_u;
3:allow ~/** r,w,s;
```

user_u is supported, too. Line 3 means home directories for user_u users.

5.3 Allow user to use sysadm_r role

You can easily allow to use sysadm_r. Open /etc/seedit/policy/sysadm_r.sp(Configuration file for sysadm_r role). add following, after *user root;*

```
user <username you want to allow to use sysadm_r>;
```

and seedit-load.

6 Creating new role

The best way to understand RBAC is to create new role. Let's see it by example. Here, we will create new role for webmaster, the name is webmaster_r. And assign the role to user whose name is webmaster.

6.1 Create uid=0 user

The uid for user that does some administration work, must be 0. It is to pass Linux permission check.
Following commands create user webmaster as uid=0.

```
# useradd -u 0 -o webmaster
# passwd webmaster
```

6.2 Create template

You can create template configuration by seedit-template command.
The usage is following.

```
seedit-template -r <role> -u <user> -o <output directory>
```

If you specify -o option, configuration is written to file, before writing to file, run command without -o option to make sure.

Following is example of generating configuration for webmaster_r role.

```
# seedit-template -r webmaster_r -u webmaster
{
role webmaster_r;
user webmaster;
include user_common.sp;
include common-relaxed.sp;
allow ~/** r,w,s;
allowpriv part_relabel;
allowpriv dac_override;
allowpriv dac_read_search;
}
```

Template configuration is generated. user webmaster can use webmaster_r role. By include common configurations to behave as login user is imported, system critical access rights are not allowed here. And webmaster_t is allowed to access user webmaster's home directory(When user webmaster login as webmaster_r he can access his home directory).
3 allowpriv are outputted, this is usually needed to do administration work.

```
allowpriv part_relabel;
```

This is necessary to use restorecon. You can use restorecon to files those webmaster_r is writable. If you do not use restorecon, delete this.

```
allowpriv dac_override;
allowpriv dac_read_search;
```

Those are necessary to skip Linux permission check.

6.3 Write to file

You can write above configuration by -o option of seedit-template.

```
# seedit-template -r webmaster_r -u webmaster -o /etc/seedit/policy/
```

Configuration is written to /etc/seedit/policy/webmaster_r.sp

6.4 Add permissions for administration

Then, add permissions to administrate web page. You need write permission under /var/www. So, add

```
allow /var/www/** r,w,s;
```

before }.

6.5 Load and test

Let's load configuration by seedit-load.

```
#seedit-load
```

And test to login. Login as webmaster. Let's see role.

```
#id
uid=0(root) gid=502(webmaster) groups=502(webmaster)
context=webmaster:webmaster_r:webmaster_t
```

Now you are webmaster!
You can upload file to /var/www/html from your home directory. If you find something is denied and it is necessary, test in permissive mode, and add configuration to /etc/seedit/policy/webmaster_r.sp using audit2spdl like normal domain.