RBAC - User Manual

Ryan DePrekel, Justin Lanyon, and Scott Linder
Secure Software Design

Table of Contents

Installation

All required files can be pulled from the github repository at https://github.com/scott-linder/rbacfs.git . To install all the dependencies required to run this program on Ubuntu or Kali based linux run the rbacDep.sh script found in /depend subdirectory as sudo.

Definitions

File access is controlled in this RBAC system by using a definitions file. An example file is shown below, in this example there are 6 users each with different roles.

```
user: role1 user1
user: role1,role2,role3 user2
user: role2 user3,user4
user: role1,role2 user5,user6

object: role1 r -r /usr
object: role2 r,x -r /lib
object: role2,role3 r,w,x /usr/file1.txt
object: role1 x /file3.txt
object: role1 x /file2.txt
```

In a definitions file you first need to define a user's role. Every line with role definition needs to start with 'user: ' which should then be followed by a space and the list of roles that/those user(s) require separated by commas. That is then followed by a space and list of users who require that role. So in our example user1 has role 1; user2 has roles 1,2, and 3; and User3 and user4 have role2.

After all the user roles are defined you need to define which roles have what access to the files or directories in your system. Every permission definition needs to start with 'object: ' which is then followed by a list of roles that will have permissions on the object separated by commas. After the roles are listed the permissions on that file need to be stated the options are r w and x; meaning read, write, and execute. You can then add the -r tag meaning recursive. This will allow the same permissions on any files within the directory defined. Finally you define the file or directory you wish to give access to; this will be defined from the root point. Do this for all files or directories in your root path.

Setup

You need to create the root filesystem that your rbac system will be based on, name it whatever you want just make sure to remember its location. This directory needs to have all directories and files mentioned in your definitions file.

Once that is complete you need to make a directory that will be used as the mounting point for the program, name it whatever you want just make sure to remember its location. This directory needs to be empty and be accessible.

Use

Once you have all your dependencies installed, your definitions file, mount point, and root point created in the /src/ directory run ‘make’, this will create the ./rbacfs executable file. The syntax for running the executable is below:

./rbacfs -o allow_other /path/to/mount /path/to/root rbac.defs

Here ./rbacfs is the executable. The ‘-o allow_other’ tag is required in order for the other users to be able to access the filesystem so it must be included in order to work. The next input needs to be your path to your empty mount directory, then followed by your path to your root and finally the path to your definitions file.