COURSEWARE

Sudoers

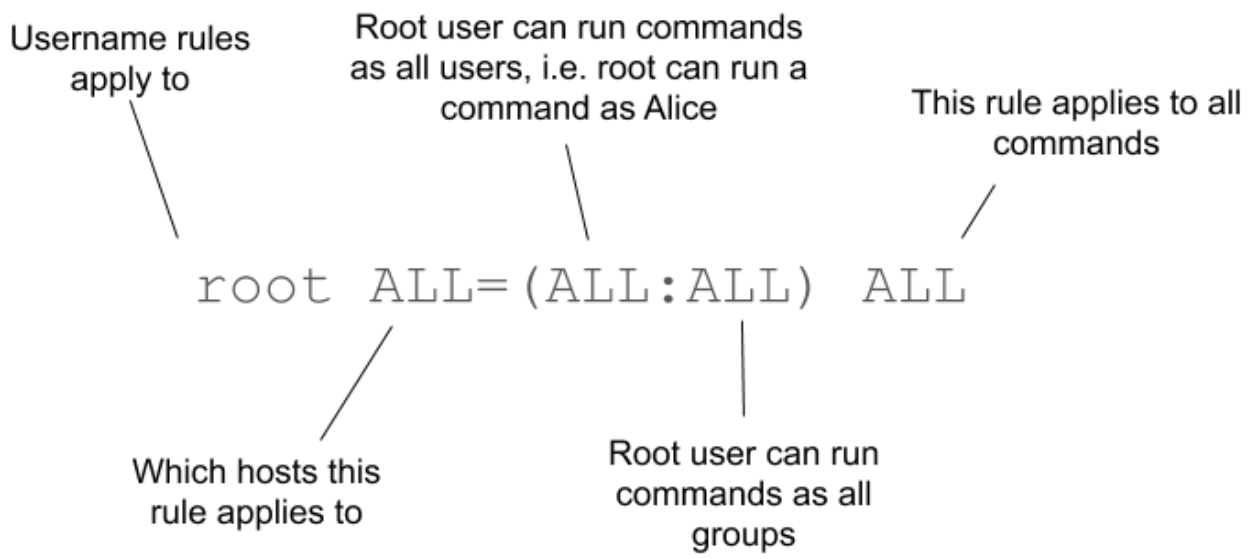
Contents

Overview

The sudo tool allows a user to act as a superuser for a command.
When you run a command as sudo, it's like running an application as administrator on a Windows machine.
Of course, only certain users can run a sudo command on a Linux machine (otherwise everyone would have administritive access!).
Users who can use sudo are configured in the `/etc/sudoers` file.
This file is extremely important and **should not be edited directly**.
If the sudoers file is broken, no one on the system can use sudo commands!
There is a tool called `visudo`, which can be used to edit the file safely.
When you save the file using `visudo`, the syntax will be checked first to make sure the file isn't currupt.

Configuring a sudo User

To edit the `/etc/sudoers` file, we can run `sudo visudo` (or just `visudo` if you are the `root` user).
An entry can be made into the sudoers file using the following format:



Run sudo Commands Without a Password

By default, a sudo user needs to enter their password when running a command as sudo.
However, this can be an issue if the commands are being run in a script.
To get around this, a sudo user can be configured to use sudo without a password:

```
# allow the user bob to run any command as sudo without a password
bob ALL=(ALL:ALL) NOPASSWD:ALL
```

Only Allow Specific Commands

Allowing all commands for a user basically just makes them a proxy root user, which often isn't what a system administrator would want.
We can allow the user to only run specific commands; for example, a Jenkins user might only need to be able to manage a systemd service, like so:

```
jenkins ALL=(ALL:ALL) NOPASSWD:\
    /bin/systemctl start nginx,\
    /bin/systemctl stop nginx,\
    /bin/systemctl status nginx
```

Make sure that you include the full path to the binaries that you are using.
For instance, the `systemctl` command's full path is `/bin/systemctl`.
To find the location of an application, you can use the `type` command:

```
type systemctl
# /bin/systemctl
```

Tutorial

These tasks will take you through configuring a `jenkins` user to manage a systemd service without using a password.
To manage a systemd service, you must be able to use the `systemctl` command; this requires elevated permissions (sudo).

Create a Jenkins User

For this example, we need a Jenkins user:

```
sudo useradd -m -s /bin/bash jenkins
```

Install NGINX

NGINX is going to be a systemd service that we will use as an example here.
Install NGINX using your relevant package manager:

```
# for ubuntu/debian use this:
sudo apt install -y nginx
# for centos/rhel use this:
sudo yum install -y nginx
```

Configure Jenkins as sudo User

To keep things simple, let's allow Jenkins to run all commands with sudo.
Start editing the `/etc/sudoers` file, by running `sudo visudo`, then enter the following into the file:

```
jenkins ALL=(ALL:ALL) NOPASSWD:ALL
```

Check the Basic sudoers Configuration Works

Let's see if it worked, by switching to the Jenkins user and running a command with sudo:

```
sudo su - jenkins
sudo echo "Hello I'm jenkins using sudo!"
exit
```

Configure sudoers to Only Manage NGINX

We can now be more specific with our sudoers configuration, by only allowing the `jenkins` user to stop, start and check the status of the NGINX systemd service.

To understand what to put in the sudoers file, let's see what commands `jenkins` will need to be able to run:

```
sudo systemctl start nginx
sudo systemctl stop nginx
sudo systemctl status nginx
```

So, jenkins will need to be able to execute the `systemctl` command.
In the sudoers file, we will need to include the full path of this binary; to do that, we can use the `type` command to find out where it is on the filesystem:

```
type systemctl
```

Now we know the entries in the sudoers file are going to have to look more like the following:

```
/bin/systemctl start nginx
/bin/systemctl stop nginx
/bin/systemctl status nginx
```

Now we can run `sudo visudo` and change the entry for jenkins to be like this:

```
jenkins ALL=(ALL:ALL) NOPASSWD:\
    /bin/systemctl start nginx,\
    /bin/systemctl stop nginx,\
    /bin/systemctl status nginx
```

Check the More Advanced Configuration Works

We can check the new sudoers entry works by switching to jenkins user and running the commands we entered:

```
sudo su - jenkins
sudo systemctl stop nginx
sudo systemctl start nginx
sudo systemctl status nginx
exit
```

Exercises

There are no exercises for this module.