Managing Remote Hosts

Basic Requirements

On this page, I'll describe how to configure Ansible to manage a remote host. In the context of this page, a `controller` is the ansible node that executes commands on remote hosts, while a `client` is a host which accepts commands from some controller.

The requirements for running / creating an ansible controller for a set of clients is as follows -

- **controller**
  - has ansible
  - create ssh key as the ansible user
    - `ssh-copy-id <worker>`
    - should be able to ssh with no password - `ssh <host / IP>` as ansible user
      - If the above does not work, create `/home/USER/.ssh/config` and add `IdentityFile /path/to/Private.key`, this will pass the key automatically when connecting as USER.
      - Ensure the host you are connecting to has the connecting key within the `~/.ssh/authorized_keys` file.
      - restart sshd.service - `sudo systemctl restart sshd.service`
- **client**
  - has ansible
  - has a known password, but can sudo without one.
    - `<user> ALL=(ALL:ALL) NOPASSWD:ALL` within sudoers

Creating a Controller

This section will configure a new user to be our Ansible controller -

- **controller**
  - has ansible
  - create ssh key as the ansible user
  - `ssh-copy-id <worker>`
  - should be able to ssh with no password - `ssh <host / IP>` as ansible user
    - If the above does not work, create `/home/USER/.ssh/config` and add `IdentityFile /path/to/Private.key`, this will pass the key automatically when connecting as USER.
    - Ensure the host you are connecting to has the connecting key within the `~/.ssh/authorized_keys` file.
    - restart sshd.service - `sudo systemctl restart sshd.service`

First, install Ansible -

```
sudo apt-add-repository --yes --update ppa:ansible/ansible && sudo apt update -y
sudo apt install software-properties-common -y && sudo apt install ansible -y
```

Creating Controller Ansible User

On the controller we plan to use to manage remote hosts, create a user that will carry out all Ansible commands.

```
sudo adduser username
[sudo] password for admin:
ssh-rsa AAAeAB3NXyXeAAADAQABAAABXwxAQDXndHlHw2DxXMk1thdTsSJWoRxXXGl5jXXMGaRta1sdprzg/sXJAdding
user `username' ...
Port 22
Adding new group `username' (1000) ...
Adding new user `username' (1000) with group `username' ...
Creating home directory `/home/username' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for username
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
```

Controller Sudo Configuration

Now that we created our user, we need to configure sudo, add `user ALL=(ALL:ALL) ALL` to the following file -

Add or edit our custom sudoers config to allow for sudo with no password

```
sudo visudo -f /etc/sudoers.d/mySudoers
```

Add the following line -

```
kansible ALL=(ALL:ALL) NOPASSWD:ALL
```

```
# Add our new user to sudo group
admin@server:~$ sudo vigr
You have modified /etc/group.
You may need to modify /etc/gshadow for consistency.
Please use the command 'vigr -s' to do so.
admin@server:~$ sudo vigr -s
You have modified /etc/gshadow.
You may need to modify /etc/group for consistency.
Please use the command 'vigr' to do so.
```

Secure the new user's User / Group ID's by defining a custom user and group ID

```
sudo usermod -u 61182 username
sudo groupmod -g 61181 username
```

Change file permissions created when we added the user. Here we are just updating user files to reflect new IDs. Errors are ok

Change all files to reference the correct group -

```
sudo find / -group 1000 -exec chgrp -h username {} \;

find: ‘/proc/18580/task/18580/fd/6’: No such file or directory
find: ‘/proc/18580/task/18580/fdinfo/6’: No such file or directory
find: ‘/proc/18580/fd/5’: No such file or directory
find: ‘/proc/18580/fdinfo/5’: No such file or directory
```

Change all files to reference the correct user -

```
sudo find / -user 1000 -exec chown -h username {} \;

find: ‘/proc/18611/task/18611/fd/6’: No such file or directory
find: ‘/proc/18611/task/18611/fdinfo/6’: No such file or directory
find: ‘/proc/18611/fd/5’: No such file or directory
find: ‘/proc/18611/fdinfo/5’: No such file or directory
```

That's it! Further customization for managing our remote servers will take place in defining hosts in the Ansible inventory, creating playbooks, and defining / applying roles. For now, we should configure a remote host to accept commands from this new Ansible controller

Creating Ansible Clients

Below, we configure a user to authenticate with on the remote host we want to admin, known as our Ansible client -

- **client**
  - has ansible
  - has a known password, but can sudo without one.
    - `<user> ALL=(ALL:ALL) NOPASSWD:ALL` within sudoers

To create an Ansible client, you'll need a user with a known password that can sudo without one. Also, we will need to install our publickey from the controller we created above into this users `~/.ssh/authorized_keys` file so Ansible can ssh and sudo on this worker with only a private key.

First, install Ansible on the remote client -

```
sudo apt-add-repository --yes --update ppa:ansible/ansible && sudo apt update -y
sudo apt install software-properties-common -y && sudo apt install ansible -y
```

Creating Ansible User for Remote Client

To speed this up, I used a script I wrote to create a user with a custom userID, and configure sudo. Get it here, or manually create the user as I did above for the Ansible controller.

If we run the script with no arguments, we see the help text -

```
sudo ./adduser.sh ansible
Illegal number of parameters.
Usage: sudo ./adduser.sh <username> <groupid>

Available groupd IDs:
60001......61183 Unused | 65520..............65533  Unused
65536.....524287 Unused | 1879048191.....2147483647  Unused
```

So we can add a user with the following command -

```
sudo ./adduser.sh ansible 524280

Adding user `ansible' ...
Adding new group `ansible' (524280) ...
Adding new user `ansible' (524280) with group `ansible' ...
Creating home directory `/home/ansible' ...
Copying files from `/etc/skel' ...
```

```
Enter 1 if ansible should have sudo privileges. Any other value will continue and make no
changes
1

Configuring sudo for ansible...

Enter 1 to set a password for ansible, any other value will exit with no password set
1

Changing password for ansible...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Configure Sudo for Remote Client

Now, we need to configure the Sudoers file to allow our user to sudo without the password, even though we did configure a password during user setup.

```
sudo visudo -f /etc/sudoers.d/mySudoers
```

Assuming your username is `ansible`, add the following line to this file. -

```
ansible ALL=(ALL:ALL) NOPASSWD:ALL
```

> Be sure to either run the `sudo visudo -f /etc/sudoers.d/mySudoers` or append the line above to the end of the default sudoers file if you ran only `sudo visudo` - This is a sequential configuration so the order of the statements is important, and we want to ensure that nothing overrides our choice to disable sudo passwords on this user

Now the ansible user can sudo with no prompt for password! Now we just need to add our controller's SSH key to the .ssh/authorized_keys file within the new ansible user's home directory.

Login as the user, and add the publickey that Ansible will pass for authentication.

```
sudo -iu username
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
username@server:~$ mkdir .ssh
username@server:~$ sudo vim .ssh/authorized_keys
```

Verify sshd_config, and restart sshd.service

```
sudo vim /etc/ssh/sshd_config
sudo systemctl restart sshd.service
```

Once you have added your key to the authorized_keys file, determine if you have or plan to have any custom PAM configurations on your host, and if so - add the following module to bypass any future changes.

Adding Listfile Module (PAM)

If you have or plan to have any custom PAM configurations on your host, you will need to change PAM sshd authentication configuration as follows to allow our user to bypass other modules

```
sudo vim /etc/pam.d/sshd
```

In `/etc/pam.d/sshd`, we can add the following line to allow for a list of users past any other modules configured on the server. Be sure to add this line at the top of our configuration file, so it is handled before any other module.

```
auth sufficient pam_listfile.so item=user sense=allow file=/etc/authusers
```

Now we can add our user to the pam_userlist.so configured in the changes made above

```
sudo vim /etc/authusers
```

In this `/etc/authusers` file, we simply list users that can bypass further PAM configurations -

```
user
otheruser
thirduser
```

Updating hosts

Be sure you add your hose IP and port to your `/etc/ansible/hosts` file, syntax is seen below -

```
[group]
www.domain.com
sub.domain.com:22
0.0.0.0
127.0.0.1:22

[othergroup]
```

```
sub.domain.com:22
127.0.0.1:22

[nginx-server]
sub.domain.com:22

[docker-host]
127.0.0.1:22

[dev]
sub.domain.com:22
```

That's it! Now just `sudo apt install ansible` and ssh to your Ansible controller to test out the configuration.

Testing Ansible client

From this point, the user is fully configured to bypass all security settings only if the ansible controller is attempting to connect, allowing full sudo access. To test this, run the following command and look for similar output -

```
ansible dev -m ping
The authenticity of host '159.203.190.63 (159.203.190.63)' can't be established.
ECDSA key fingerprint is SHA256:jDxFV7KA00wNIdpG40ppvW2RobNXyPeItdi4jL3h78s.
Are you sure you want to continue connecting (yes/no)? yes
worker.domain.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

This test says that the host was not changed (`"changed": false`), and the server accepted our connection (`"ping":"pong"`)

Revision #23
Created 2019-07-28 08:17:22 UTC by Shaun Reed
Updated 2020-08-03 00:18:17 UTC by Shaun Reed