Table of Contents

Ansible with Semaphore 3
Installation 3
A classic ¡Hola Mundo! 5
Client configuration 12
Execute the task 13
Automatic start with systemd 15
Configure HTTPs with Apache 15
Local git repository 16
FreeIPA or RedHat IdM 16
References 17

Ansible with Semaphore

A brand new honest howto about Ansible with Semaphore.

Tested in a **clean installation of** CentOS 7.x, must works in RHEL 7.x and Oracle 7.x.

Installation

Check download link of Semaphore from: https://github.com/ansible-semaphore/semaphore/releases and replace <link>

```
$ curl -L
https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm -o
/tmp/epel-release-latest-7.noarch.rpm
$ sudo yum localinstall /tmp/epel-release-latest-7.noarch.rpm
$ sudo yum update
$ sudo yum install mariadb-server git ansible
$ sudo curl -L <link> -o /usr/bin/semaphore
$ sudo chmod +x /usr/bin/semaphore
$ sudo useradd semaphore
$ sudo passwd semaphore
```

Configure mariadb-server, if you don't have it installed:

```
$ sudo systemctl start mariadb
$ sudo systemctl enable mariadb
$ sudo mysql_secure_installation
$ sudo firewall-cmd --zone=public --add-port=3000/tcp --permanent
$ sudo firewall-cmd --reload
```

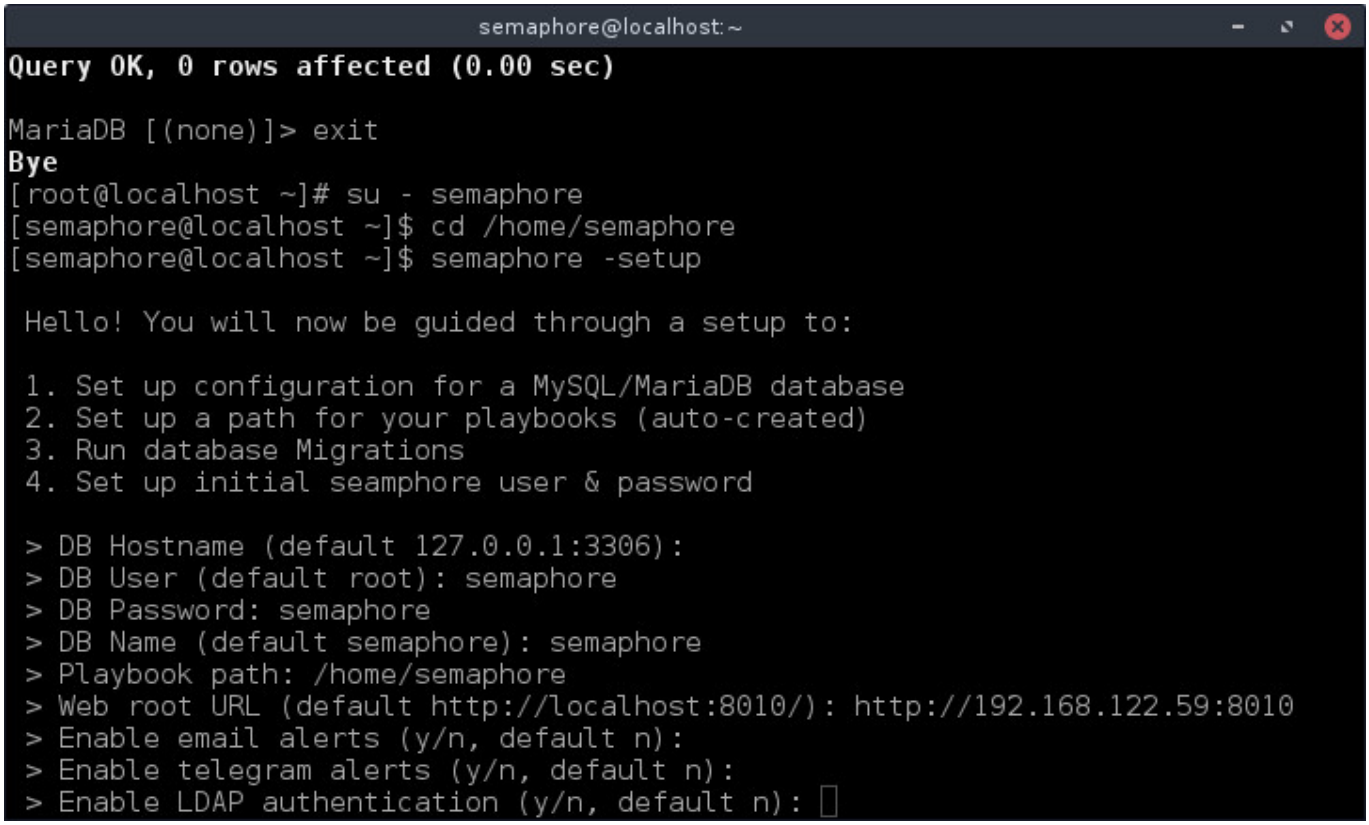
As root user create a database, change username and password if you want:

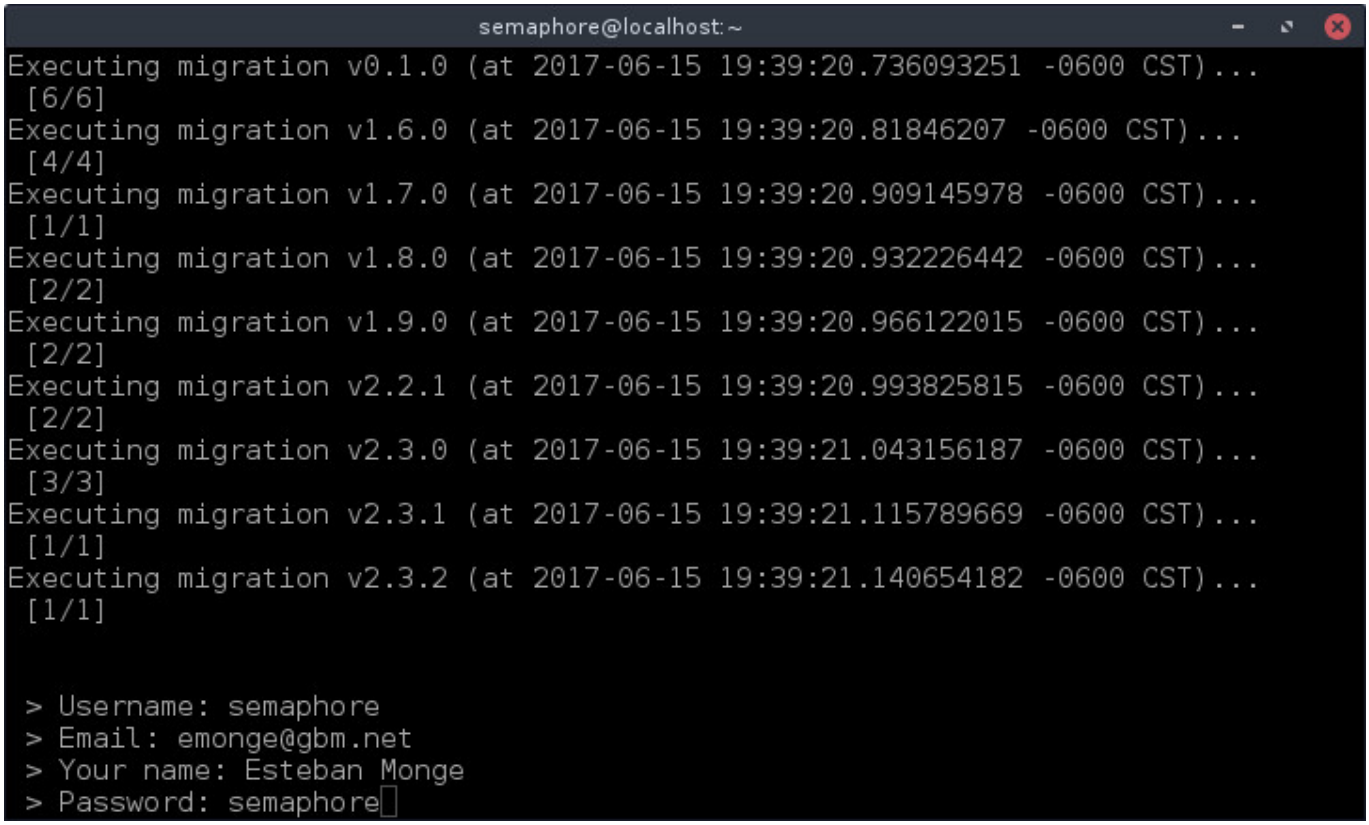
```
$ mysql -u root -p
$ MariaDB [(none)]> create database semaphore;
$ MariaDB [(none)]> grant all privileges on semaphore.* to
semaphore@localhost identified by 'semaphore';
$ MariaDB [(none)]> exit
```

Start and configure the server as semaphore user:

```
$ su - semaphore
$ cd /home/semaphore
$ semaphore -setup
```

The wizard will guide you to configure all data about database and temporal folder, you need to configure /home/semaphore or any folder writable by semaphore user.





Now you can start the server:

```
$ nohup semaphore -config /home/semaphore/semaphore_config.json &
```

With this command you can check any issue in configuration. For example:

```
$ tail -f /home/semaphore/nohup.out
```

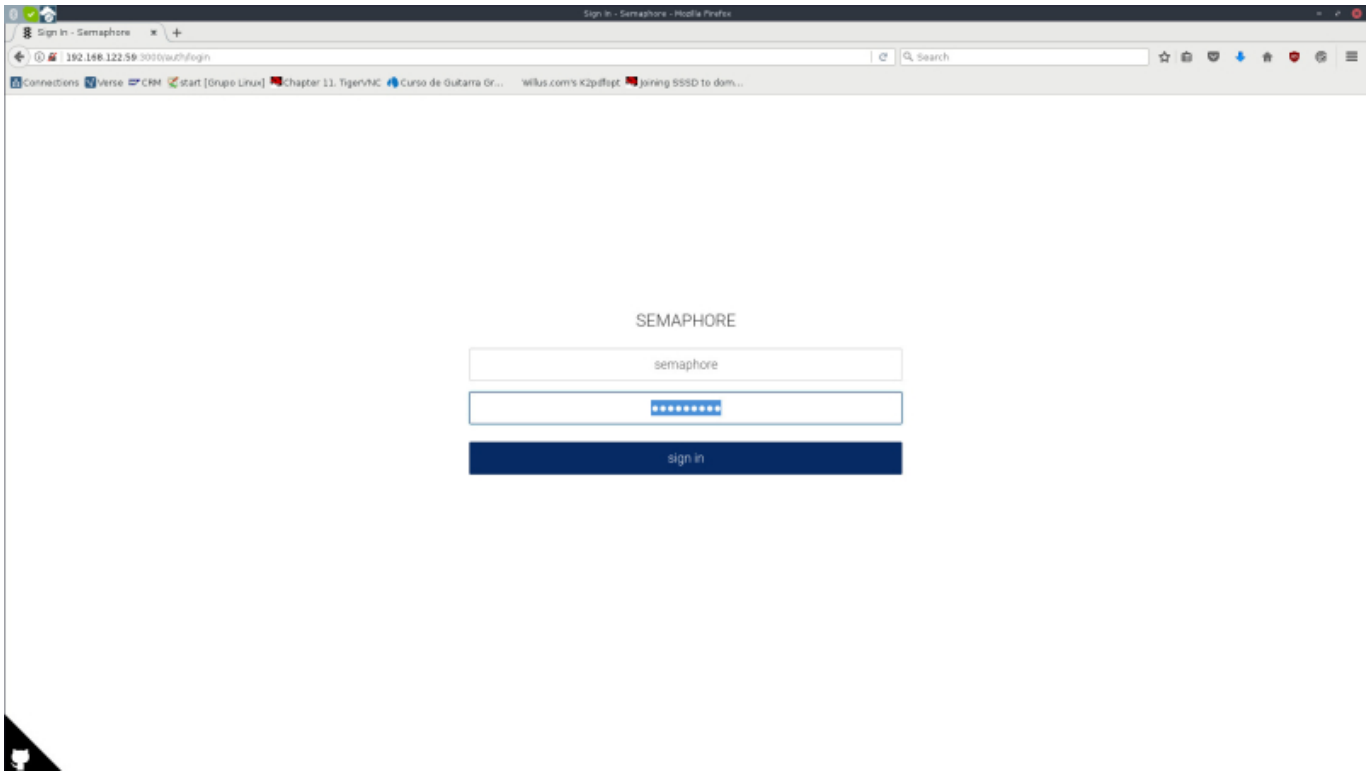
Last step is the creation of ssh key, as semaphore user, do no use password:

```
$ ssh-keygen
```

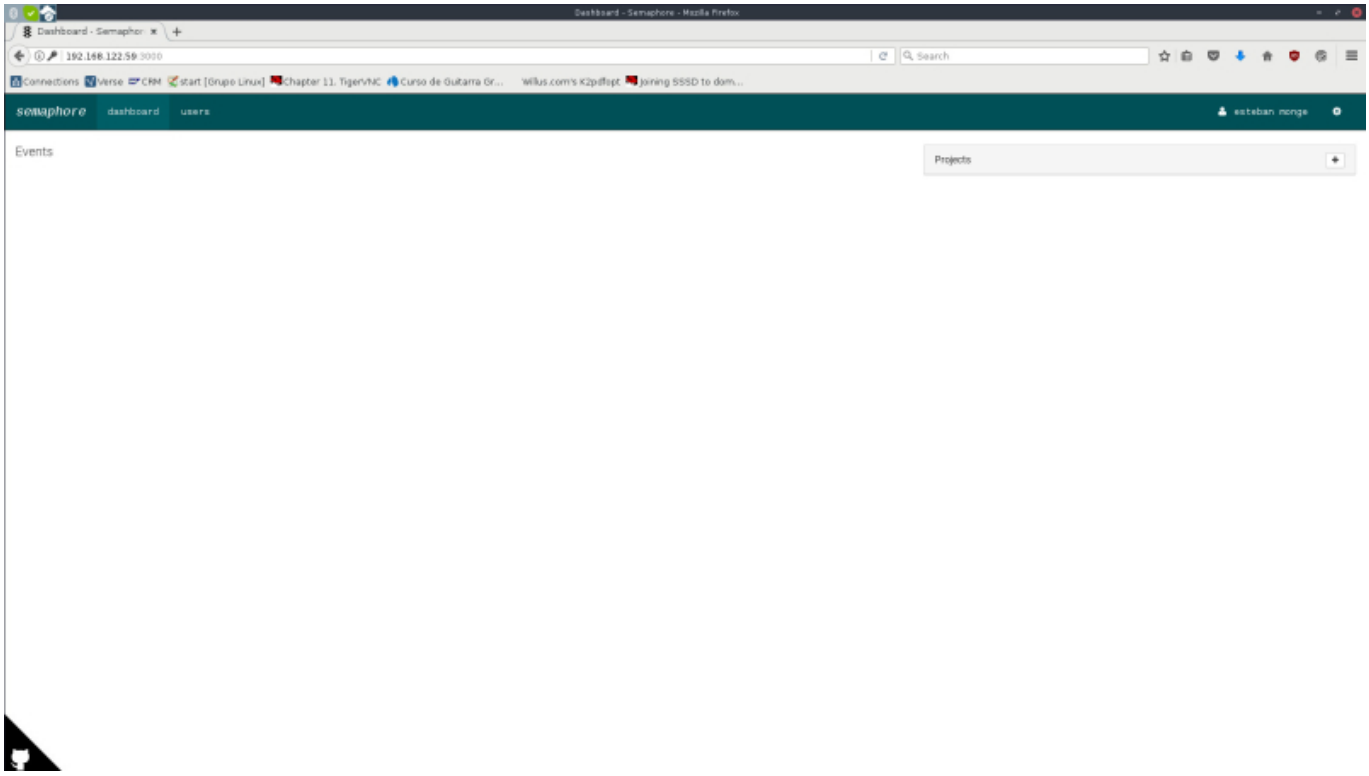
Copy /home/semaphore/.ssh/id_rsa in a safe place, you will need the content of this file to get work Ansible.

A classic ¡Hola Mundo!

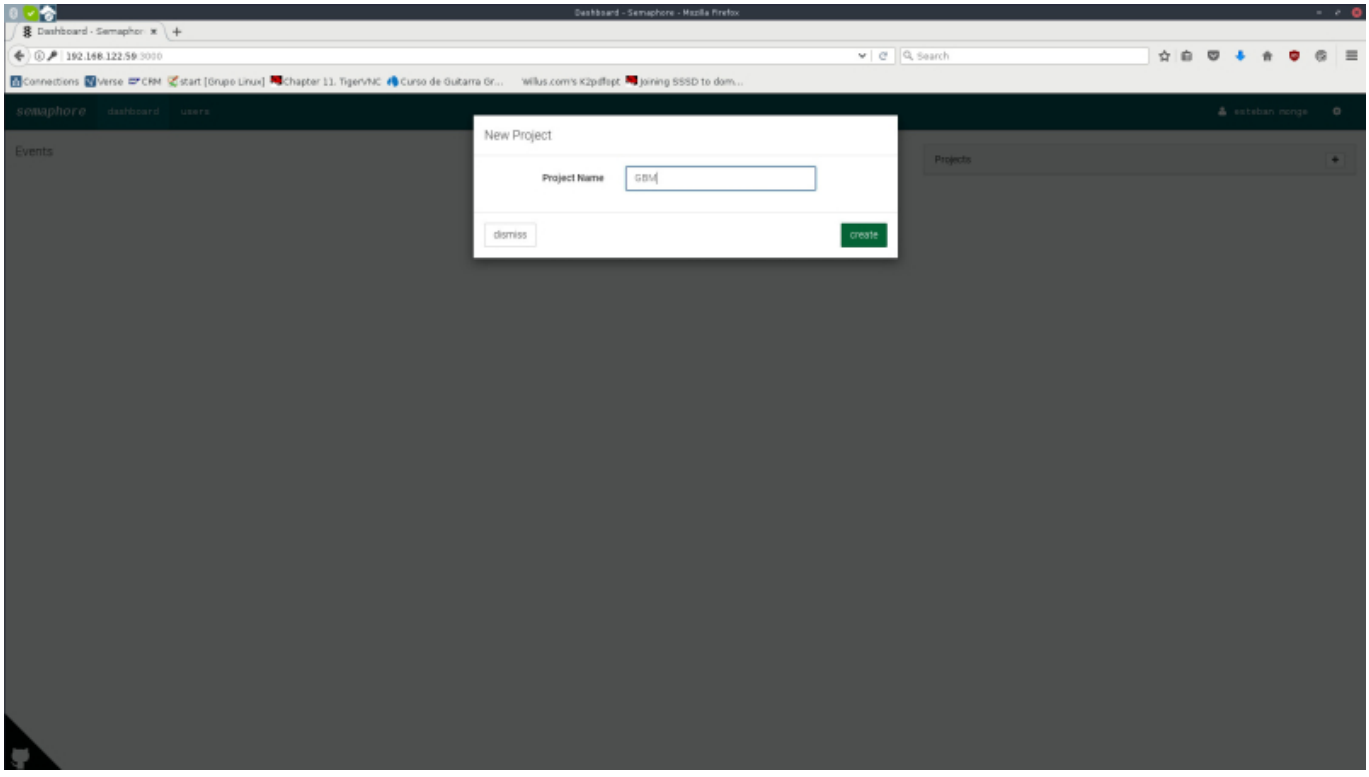
Login screen



Dashboard

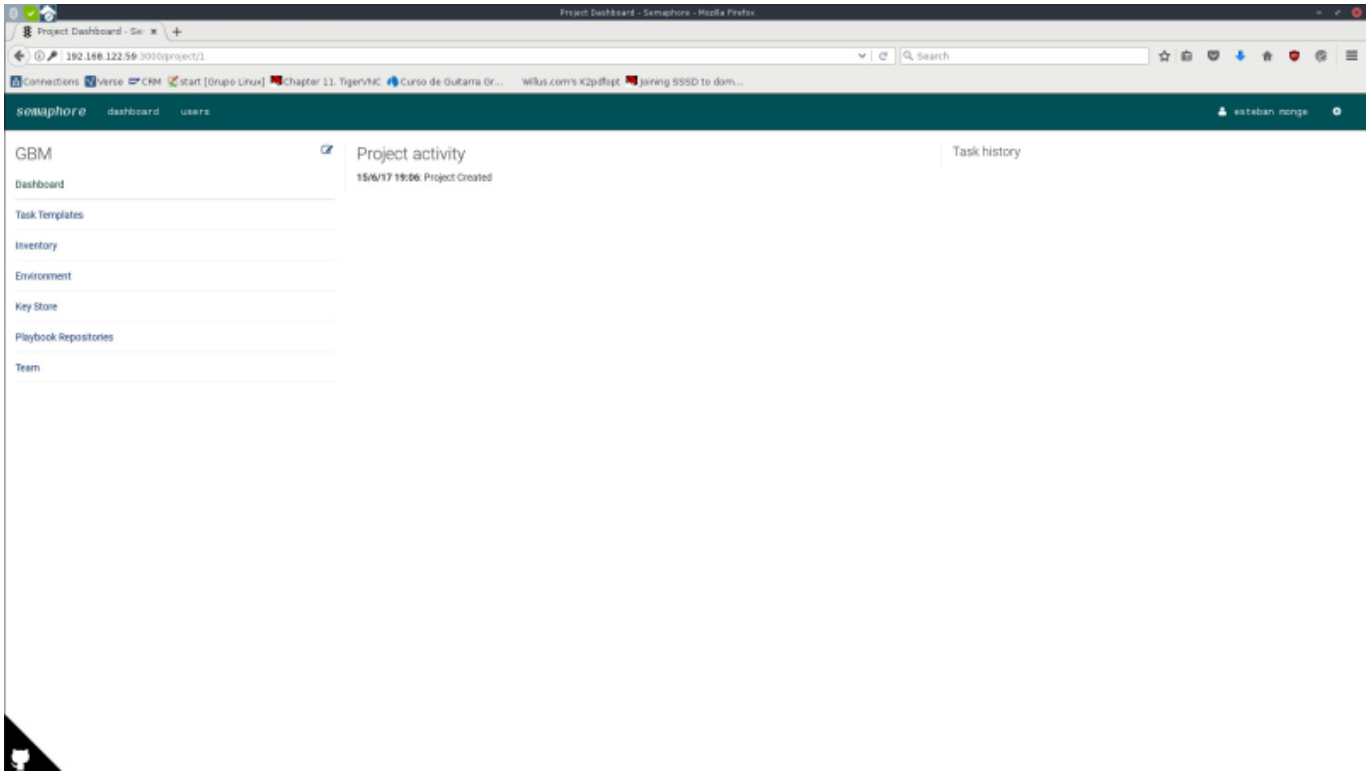


Create a new project

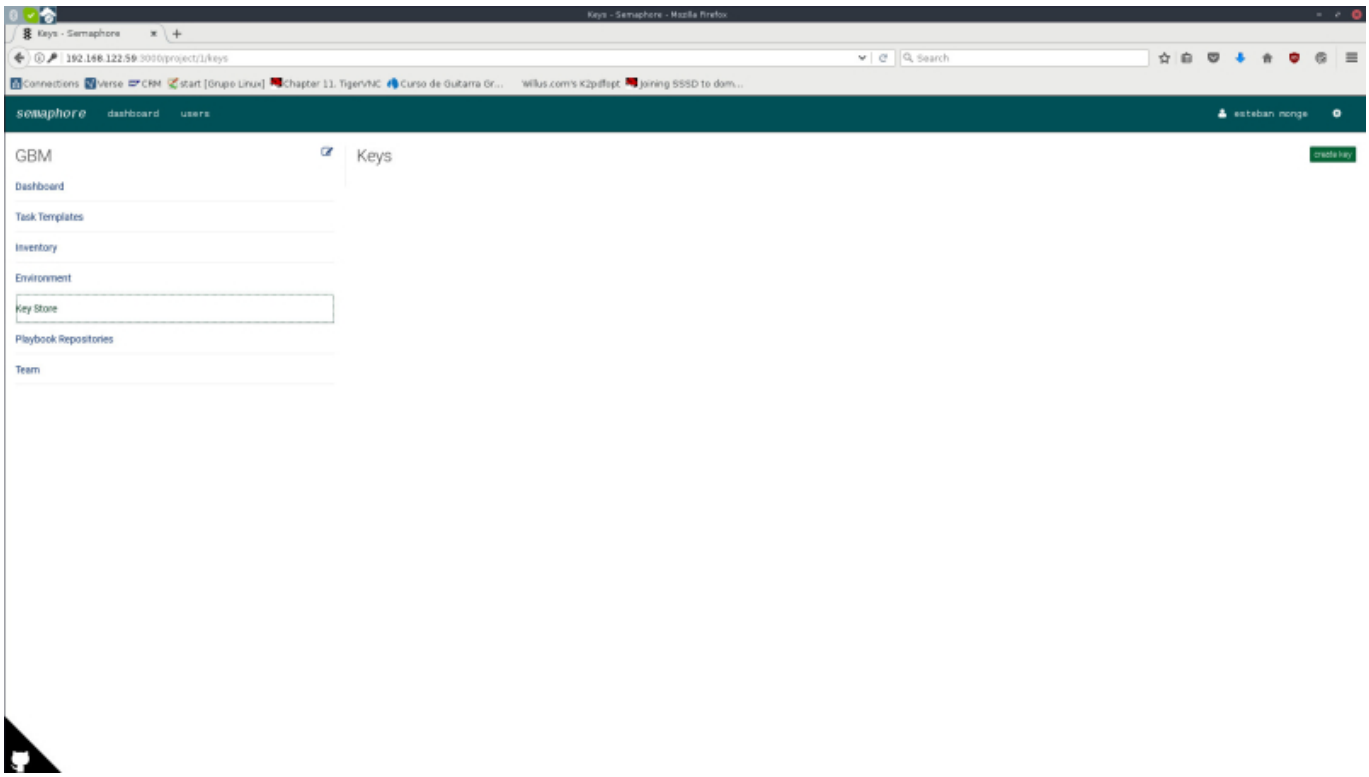


Project configuration

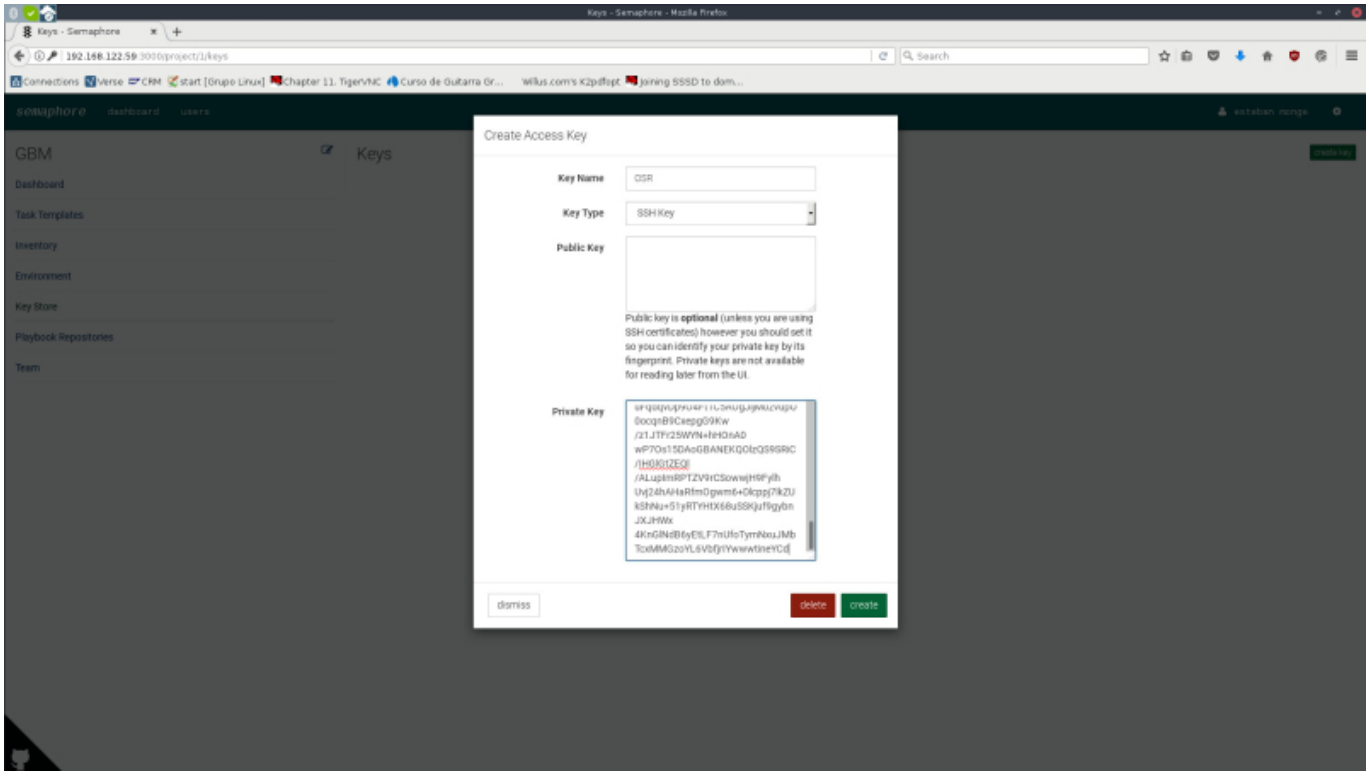
Click on project name, you will see the Project activity screen:



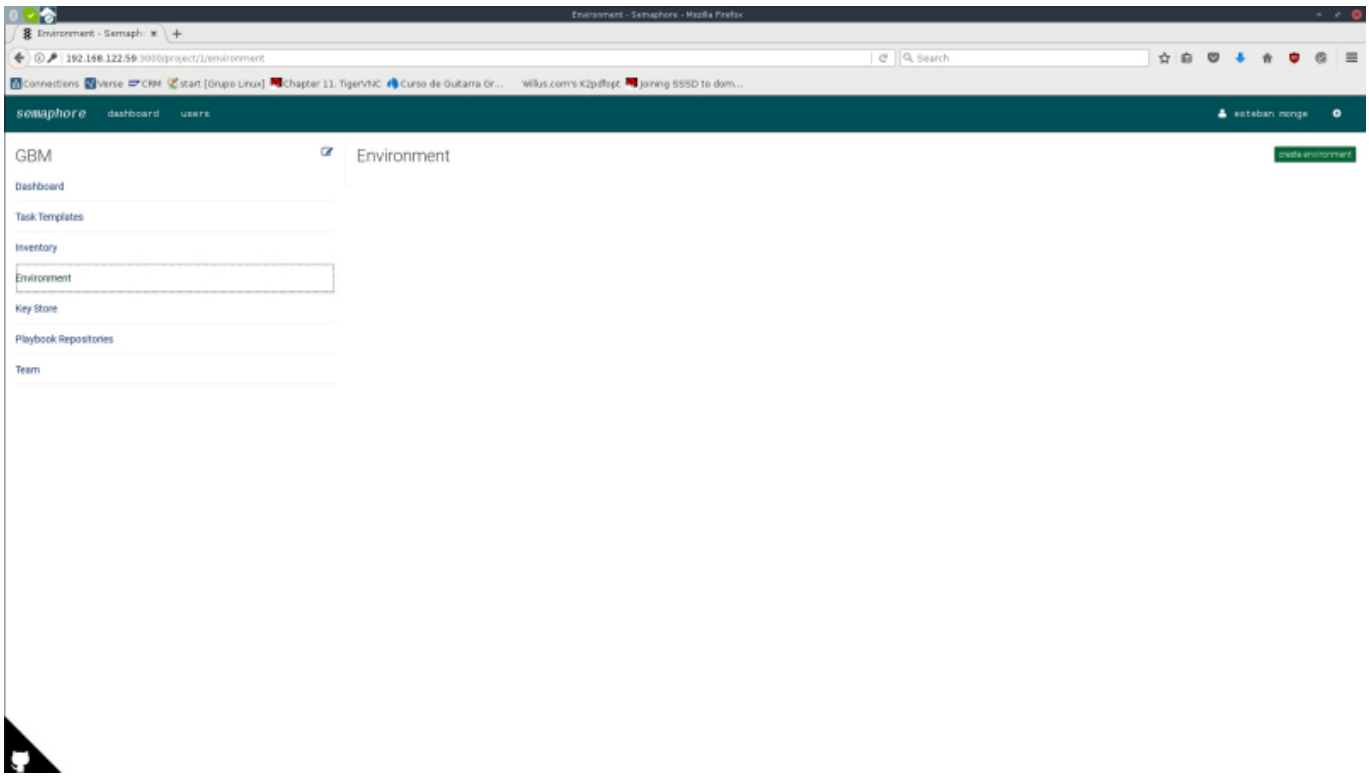
Select Key Store option:



Select create and put a name and select “SSH Key”, copy and paste the content of the file id_rsa that you extract from semaphore user.

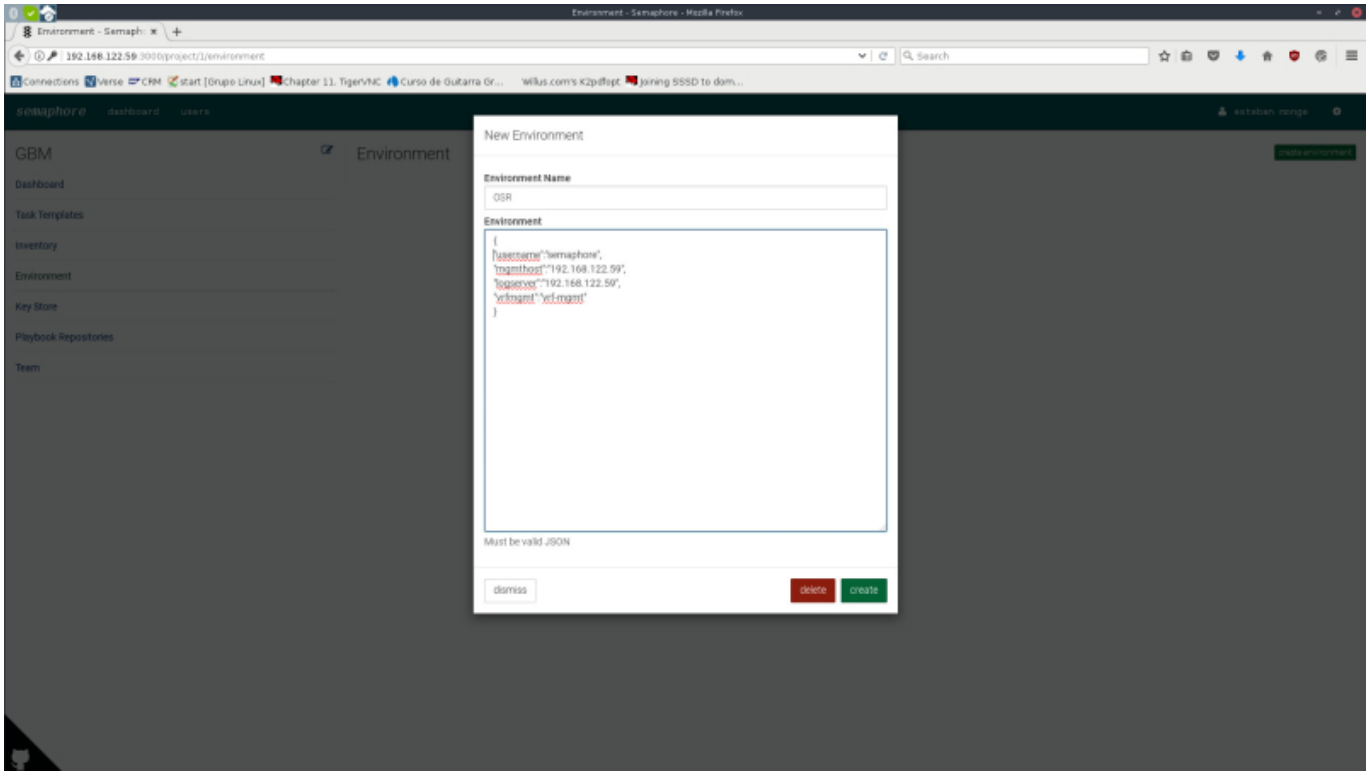


Select Environment option:

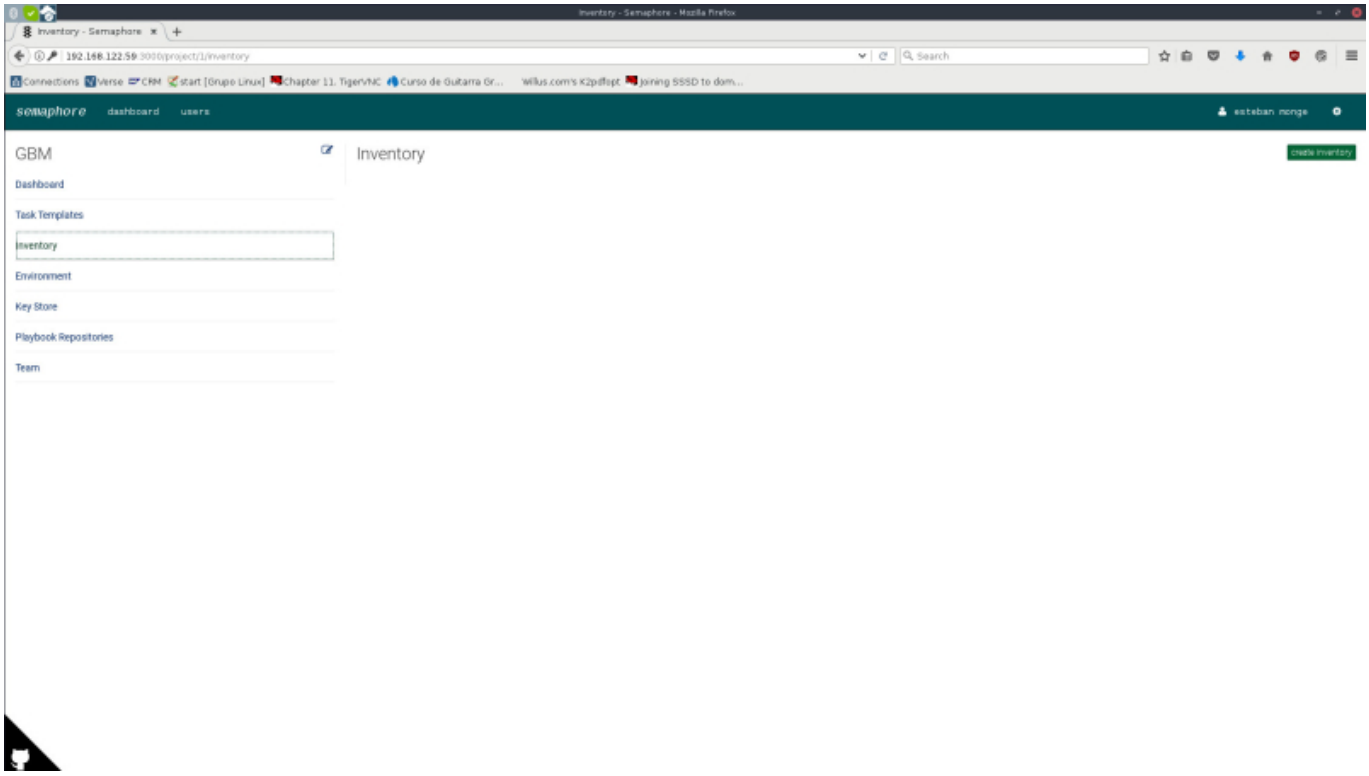


Select create and put a friendly name and paste a content similar to this:

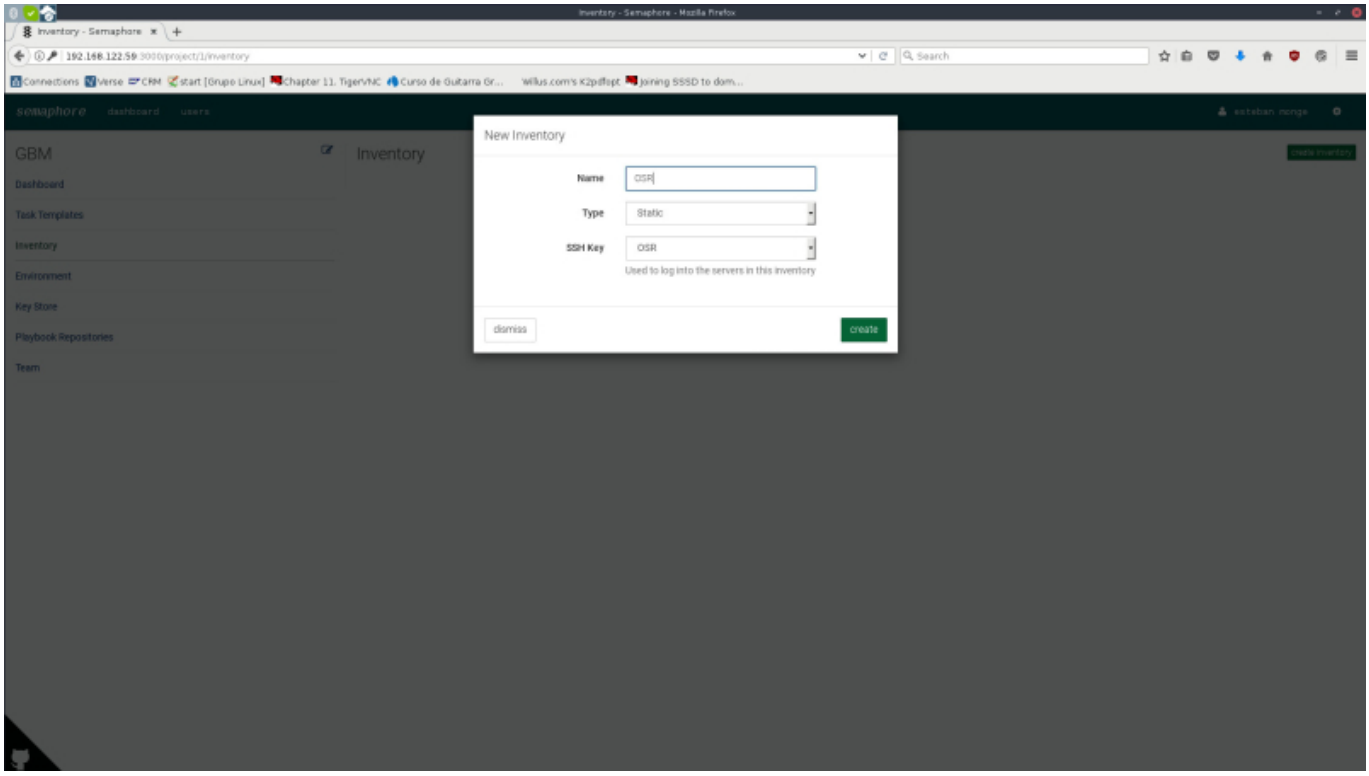
```
{
"username":"semaphore",
"mgmthost":"192.168.122.59",
"logserver":"192.168.122.59",
"vrfmgmt":"vrf-mgmt"
}
```



Select Inventory option:

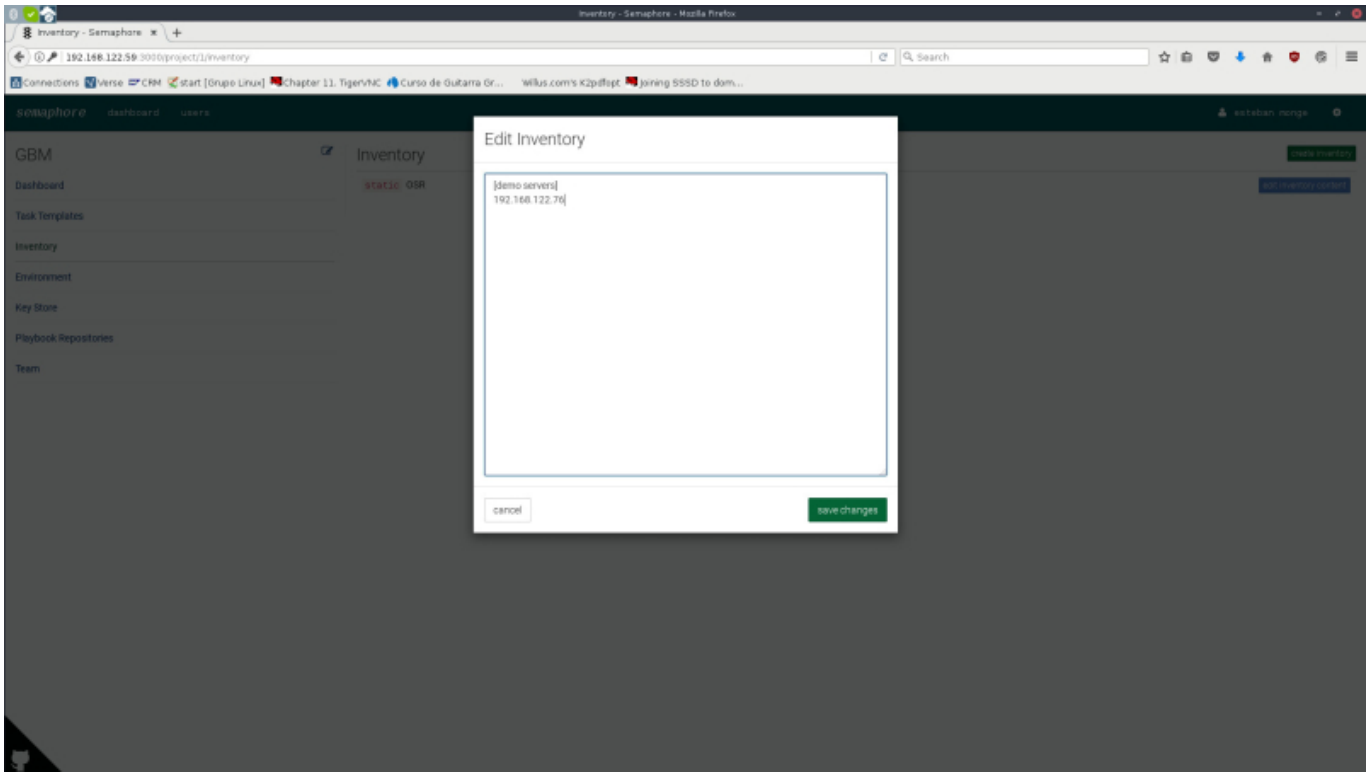


Select create and put a name, select Static in type and select the previous created SSH Key:



Select edit inventory content and add content similar to this:

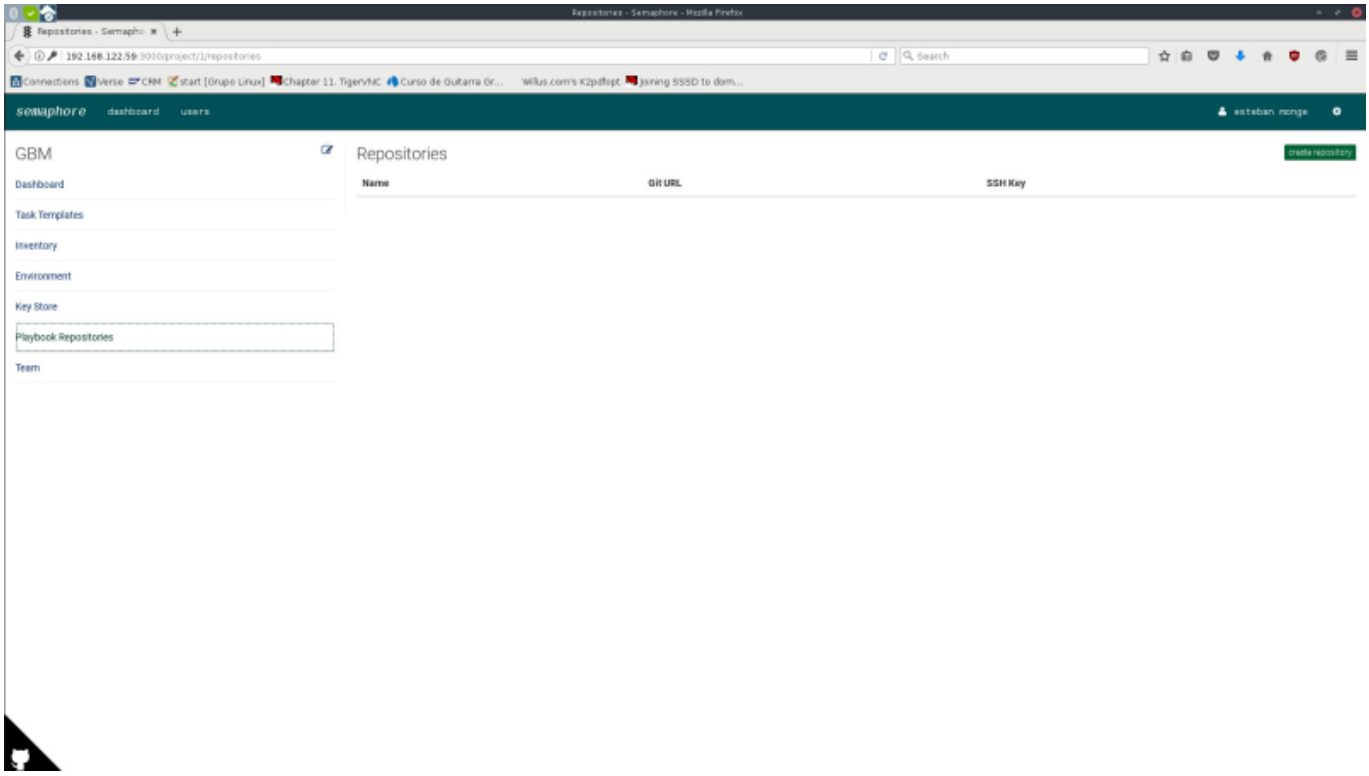
```
192.168.122.76
```



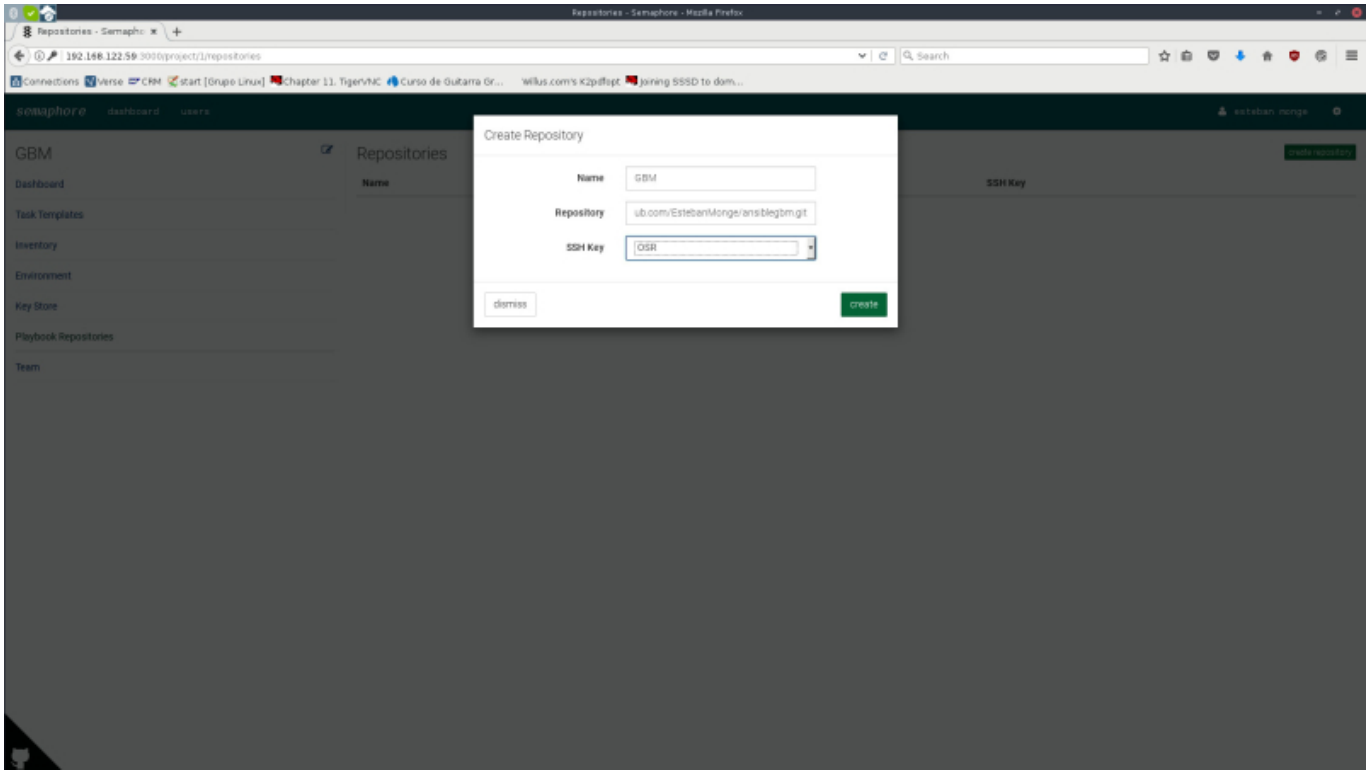
Repository configuration

You will need a git repository, for example one from www.github.com

Select Playbook Repositories option:

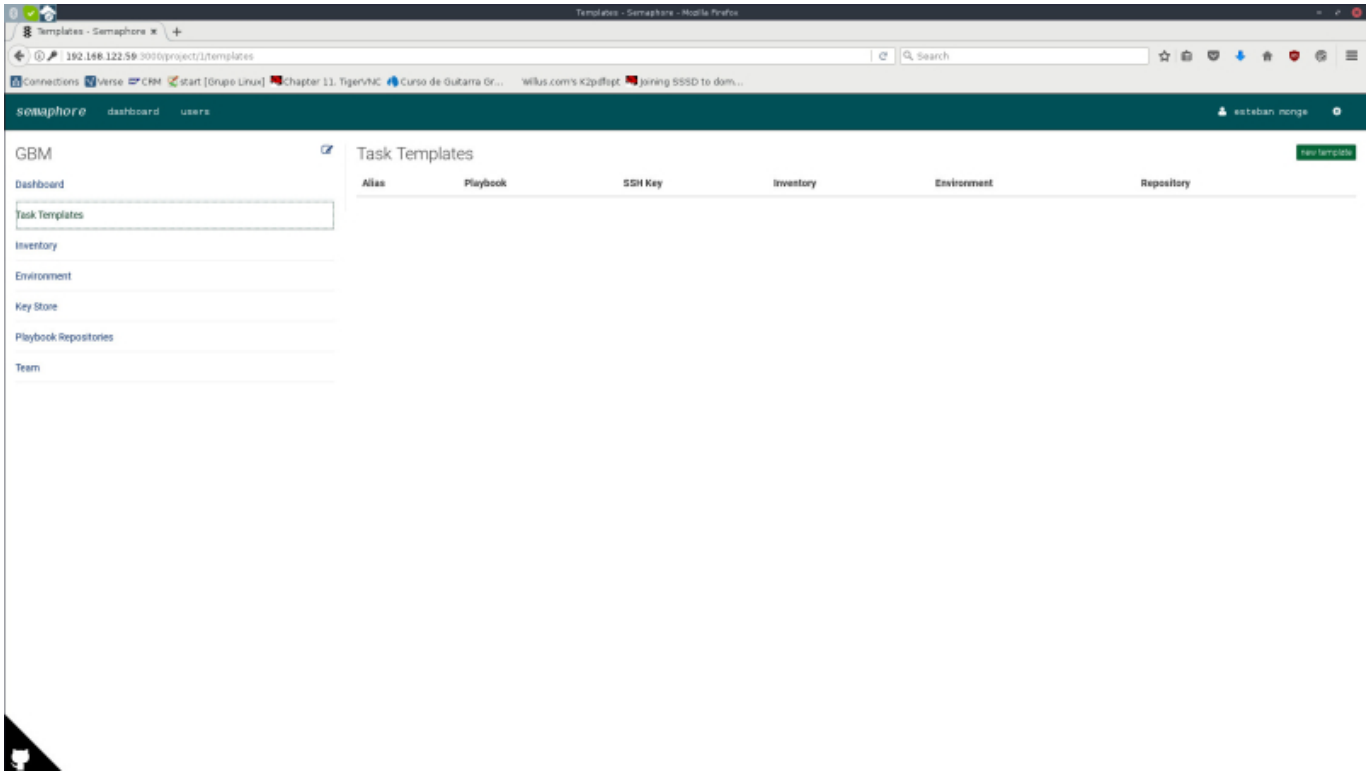


Select create repository and put a friendly name, put the URL of the repository and select one SSH Key.

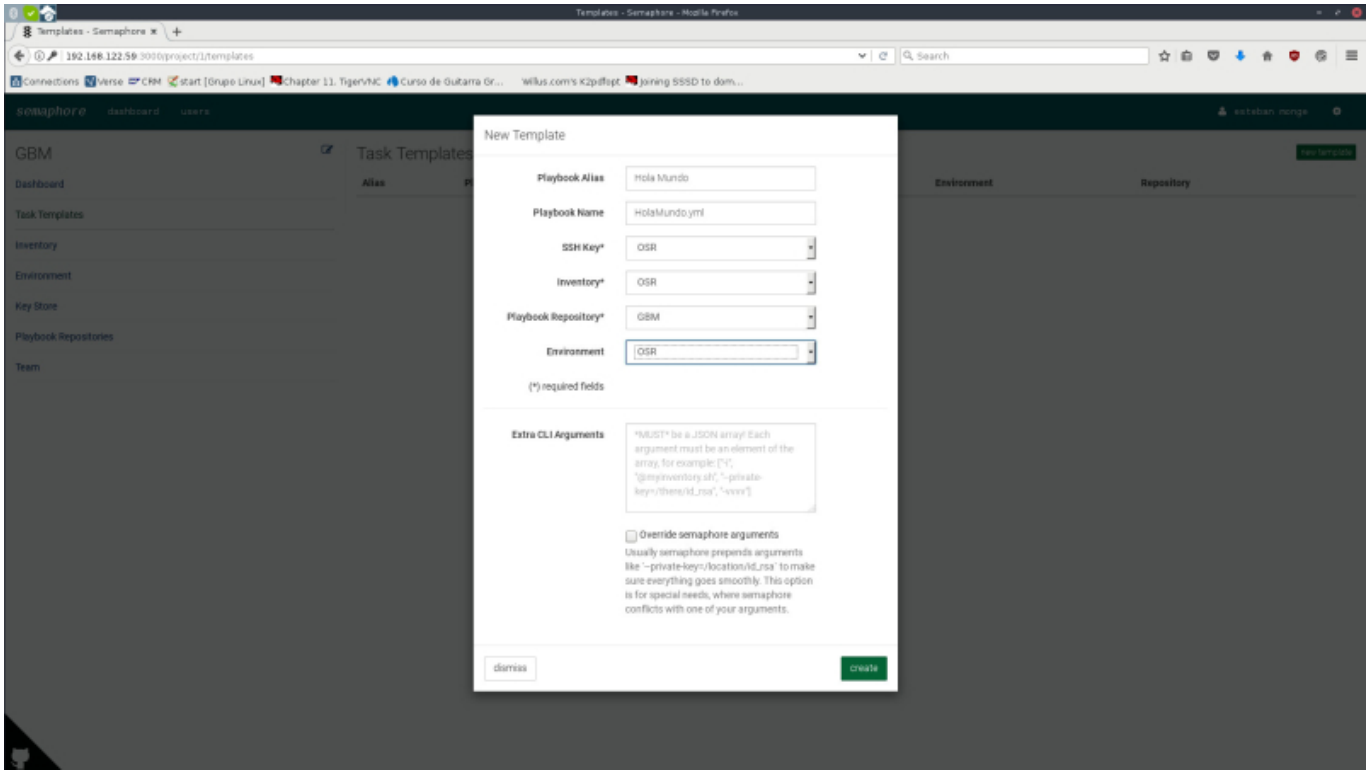


Task Templates

Select Task Templates option. You will store all task templates that will execute the automatic tasks in each server defined in the inventory.



Select new template, put a name to the task, in Playbook Name write the exact name of the file in the repository, select SSH Key, Inventory, Playbook Repository and Environment.



Client configuration

In the client create a user named semaphore:

```
$ sudo useradd semaphore
```

```
$ sudo passwd semaphore
```

The client must have Python installed, in our case CentOS, RHEL and Oracle must have installed Python.

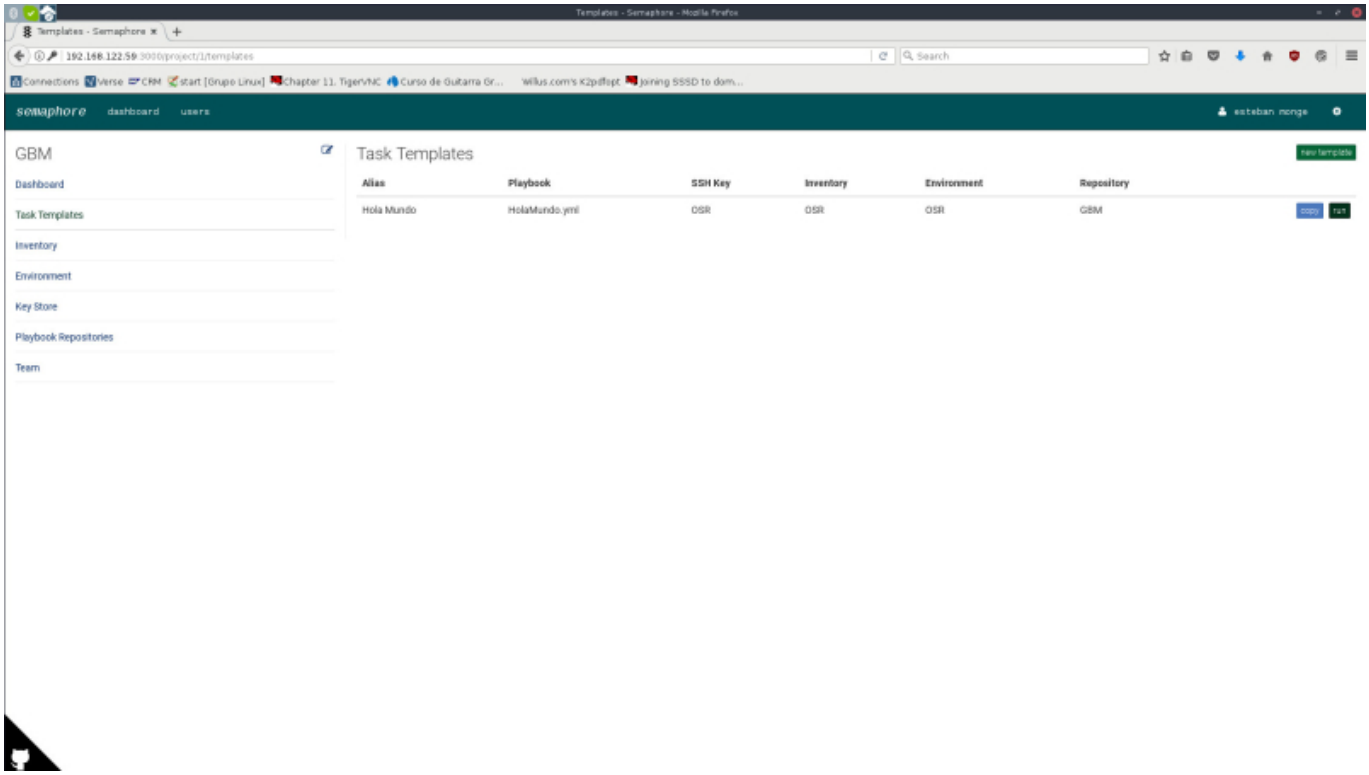
You need copy the SSH key to the client server, in our example 192.168.122.76, in the Semaphore server execute:

```
$ su - semaphore
$ ssh-copy-id semaphore@192.168.122.76
```

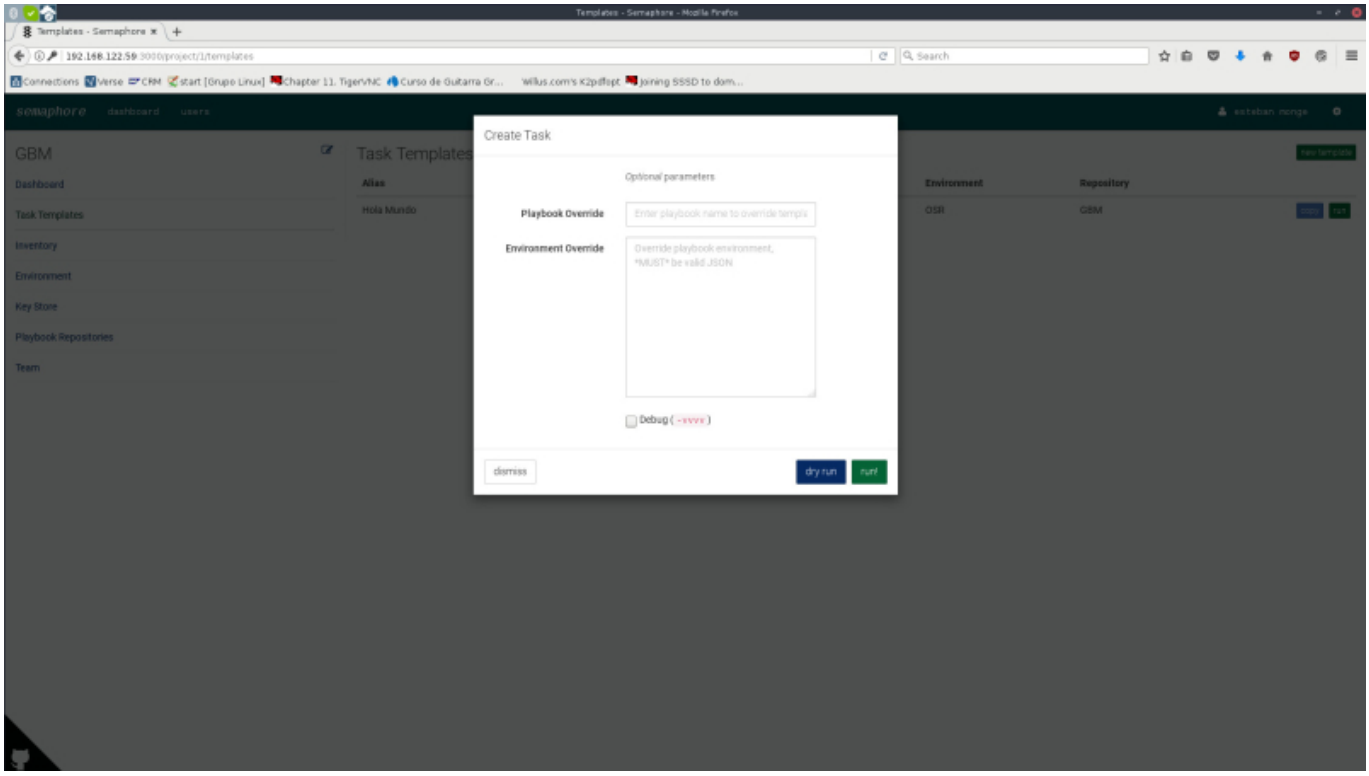
Write the password.

Execute the task

Select Task Templates option and choose run:

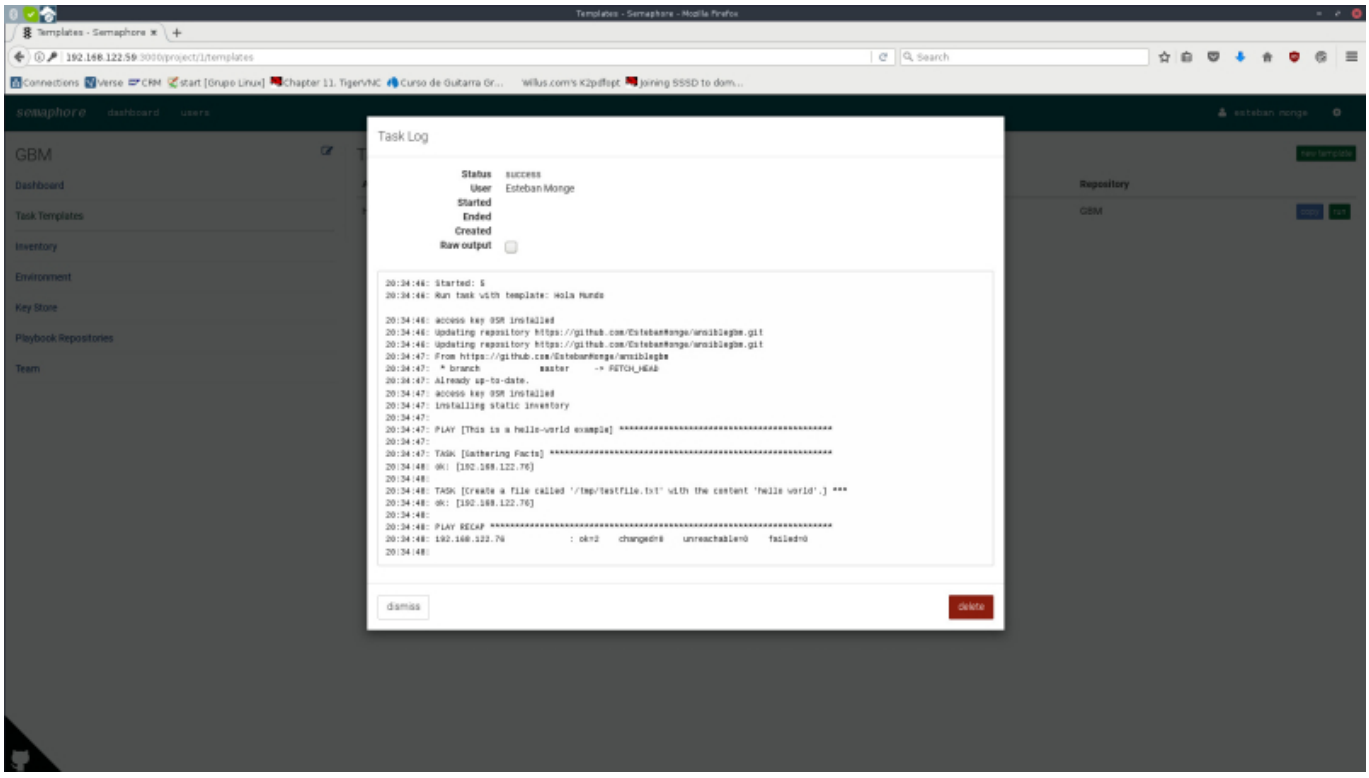


Select dry run, select this option every time that you create a new Task Template o when you change anything in the git repository.

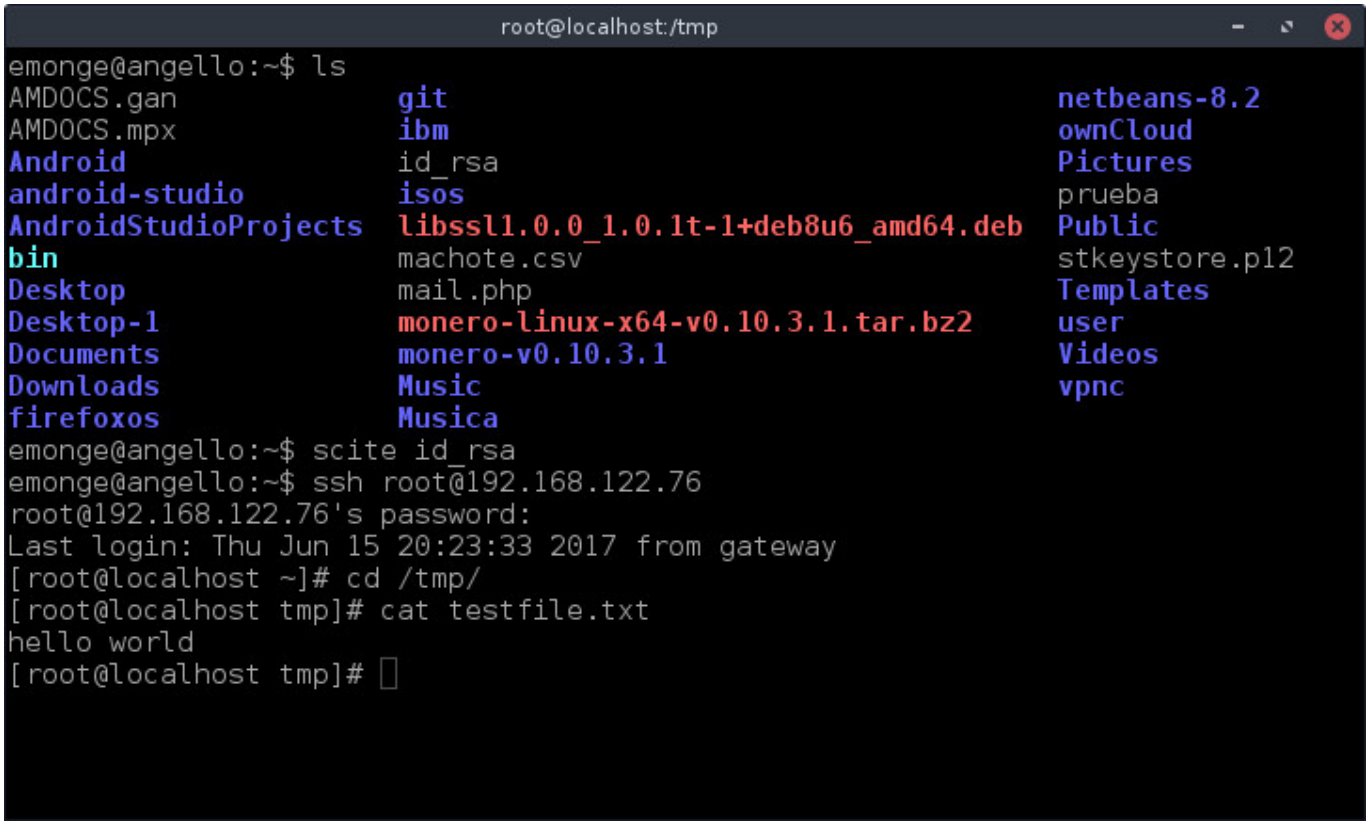


If everything is OK you will get a line similar to this in Task Log screen:

```
20:34:48: PLAY RECAP
*********************************************************************
20:34:48: 192.168.122.76                 : ok=2    changed=0    unreachable=0
failed=0
```



Also you can check that in the client server 192.168.122.76 in the directory /tmp a file named testfile.txt:



Automatic start with systemd

```
$ sudo vi /etc/systemd/system/semaphore.service
```

Put this content:

```
[Unit]
Description=Semaphore
After=network.target

[Service]
User=semaphore
Group=semaphore
ExecStart=/usr/bin/semaphore -config /home/semaphore/semaphore_config.json
Restart=always

[Install]
WantedBy=multi-user.target
```

Finally start and enable the service:

```
$ sudo systemctl start semaphore
$ sudo systemctl enable semaphore
```

Configure HTTPs with Apache

```
$ sudo yum install httpd mod_ssl
```

Edit /etc/httpd/conf.d/ssl.conf, add the following code:

```
    ProxyPreserveHost On

    ProxyPass / http://0.0.0.0:3000/
    ProxyPassReverse / http://0.0.0.0:3000/
</VirtualHost>
```

Configure SELinux if you don't have down:

```
$ sudo semanage port -a -t http_port_t -p tcp 3000
```

If you don't have semanage installed:

```
$ sudo yum install policycoreutils-python
```

Enable ports and restart Apache:

```
$ sudo firewall-cmd --zone=public --add-port=443/tcp --permanent
$ sudo firewall-cmd --reload
$ sudo systemctl restart httpd
$ sudo systemctl enable httpd
```

Check the URL, for example: https://192.168.122.59

I recommend remove from firewalld the http port:

```
$ sudo firewall-cmd --zone=public --remove-port=3000/tcp --permanent
$ sudo firewall-cmd --reload
```

Local git repository

```
$ mkdir /home/semaphore/semaphore.git
$ cd /home/semaphore/semaphore.git
$ git --bare init
$ ssh-copy-id semaphore@SERVERNAME
```

In repository put the next address:

```
semaphore@SERVERNAME:/home/semaphore/semaphore.git
```

FreeIPA or RedHat IdM

Edit semaphore_config.json, replace:

```
        "ldap_enable": true,
        "ldap_binddn": "uid=USER,cn=users,cn=accounts,dc=EXAMPLE,dc=COM",
        "ldap_bindpassword": "PASSWORD",
        "ldap_server": "SERVER:389",
        "ldap_needtls": true,
        "ldap_searchdn": "cn=users,cn=accounts,dc=EXAMPLE,dc=COM",
        "ldap_searchfilter":
"(&(uid=%s)(memberOf=cn=semaphore,cn=groups,cn=accounts,dc=EXAMPLE,dc=COM))"
,
        "ldap_mappings": {
                "dn": "dn",
                "mail": "mail",
                "uid": "uid",
                "cn": "cn"
        },
```

Replace:

- USER = username to connect LDAP
- PASSWORD = password of the USER to connect LDAP
- dc=EXAMPLE,dc=COM = FQDN
- SERVER=hostname or IP of LDAP server
- The users must belong to semaphore group (change if you want)

FreeIPA and RedHat IdM doesn't allow anonymous queries about groups, that is the reason why you need the user for connection.

References

- http://codingbee.net/tutorials/ansible/ansible-example-playbook
- https://github.com/ansible-semaphore/semaphore/wiki/First-Steps
- https://github.com/EstebanMonge/ansiblegbm
- https://techbloc.net/archives/1502
- https://www.digitalocean.com/community/tutorials/how-to-use-apache-as-a-reverse-proxy-with-mod_proxy-on-centos-7
- https://www.digitalocean.com/community/tutorials/how-to-use-apache-http-server-as-reverse-proxy-using-mod_proxy-extension
- https://www.digitalocean.com/community/tutorials/how-to-create-an-ssl-certificate-on-apache-for-centos-7