Name: Peñas, Issa Victoria H.	**Date Performed:** 10/11/2022
Course/Section: CPE232 - CPE31S22	**Date Submitted:** 10/11/2022
Instructor: Dr. Jonathan V. Taylar	**Semester and SY:** 1st Semester (SY: 2021 - 2022)

Activity 8: Install, Configure, and Manage Availability Monitoring tools

1. Objectives

Create and design a workflow that installs, configure and manage enterprise monitoring tools using Ansible as an Infrastructure as Code (IaC) tool.

2. Discussion

Availability monitoring is a type of monitoring tool that we use if the certain workload is up or reachable on our end. Site downtime can lead to loss of revenue, reputational damage and severe distress. Availability monitoring prevents adverse situations by checking the uptime of infrastructure components such as servers and apps and notifying the webmaster of problems before they impact on business.

3. Tasks

1. Create a playbook that installs Nagios in both Ubuntu and CentOS. Apply the concept of creating roles.
2. Describe how you did step 1. (Provide screenshots and explanations in your report. Make your report detailed such that it will look like a manual.)
3. Show an output of the installed Nagios for both Ubuntu and CentOS.

4. Output

1. Create a new repository in GitHub under the name of **HOA-8.1- Nagios**, and make sure that the repository is **Public.** As a good practice add a **READ.md** file and input any related information regarding your inserted repository

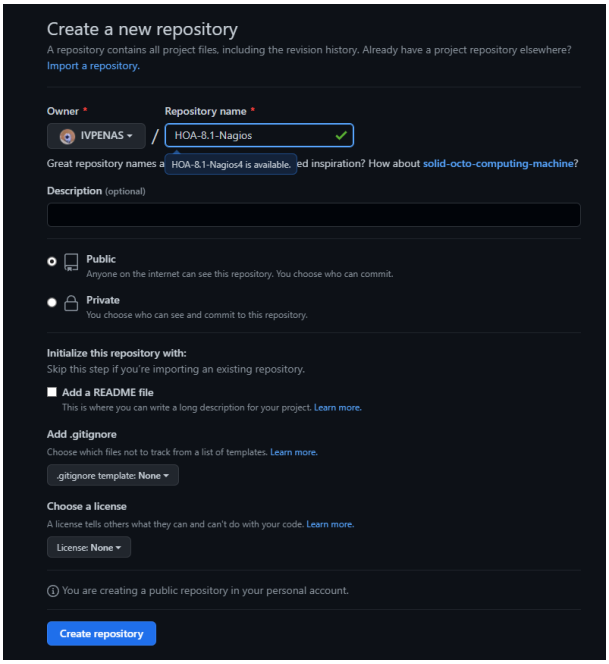


Figure 1.1. Shows the creation of New Repository in Github

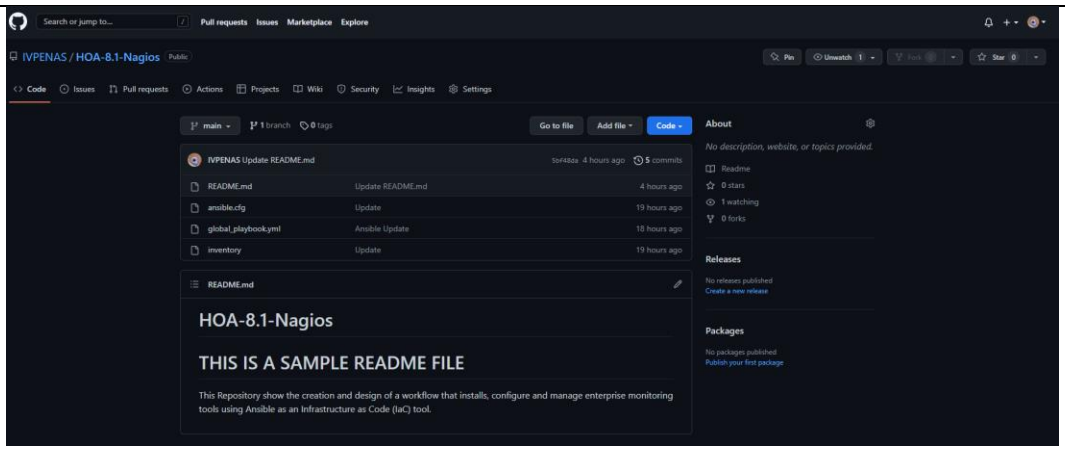


Figure 1.2. Shows the Repository (Note: I've already created it before making this Document)

2. After creating the new repository make sure that the Local Server was connected to it using the command **git clone [ssh link]** whereas the ssh link can be found inside the **code** button

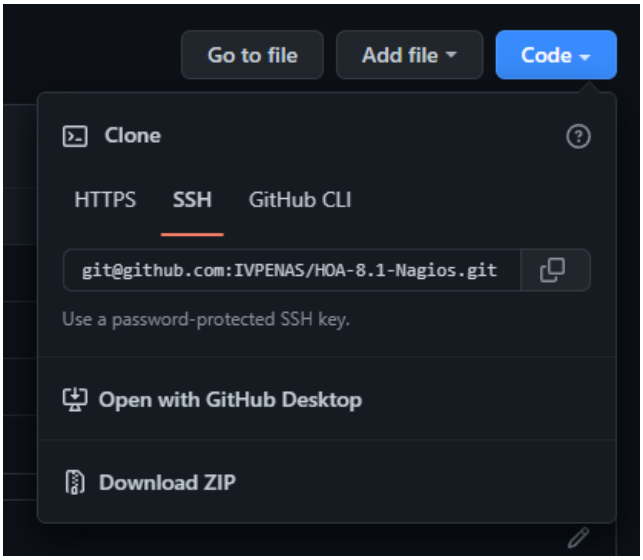


Figure 1.3. Generating a SSH Link to connect it to the servers

```
penas@penas-VirtualBox:~$ git clone git@github.com:IVPENAS/HOA-8.1-Nagios.git
Cloning into 'HOA-8.1-Nagios'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 27 (delta 2), reused 20 (delta 1), pack-reused 0
Receiving objects: 100% (27/27), 4.68 KiB | 4.68 MiB/s, done.
Resolving deltas: 100% (2/2), done.
penas@penas-VirtualBox:~$ ls
 Desktop     HOA-8.1-Nagios          Pictures   Templates
 Documents  'HOA-8.1-Nagios(copy)'   Public     Videos
 Downloads   Music                   snap
penas@penas-VirtualBox:~$
```

Figure 1.4. Cloning the Repository to the Local Machine which serves our Folder

3. Change your directory on the cloned folder from GitHub using **cd CE232_penas/Nagios** and create the base ansible structure **ansible.cfg** and **inventory** whereas you insert the selected Servers, in my case I chose Server 1 - 192.168.56.112 and CentOS - 192.168.56.111. and prepare a **.yml file** for the global configuration playbook for the whole ansible.

Note: In this case the created directories were made during face-to-face classes which in later parts, you'll see different names of directories yet same file names except the main base .yml file which will be named as global_playbook.yml

```
penas@penas-workstation-VirtualBox:~/CPE232_penas/Nagios$ tree
.
├── ansible.cfg
├── inventory
└── main_playbook.yml

0 directories, 3 files
penas@penas-workstation-VirtualBox:~/CPE232_penas/Nagios$ cat ansible.cfg
[defaults]
inventory = inventory
host_keychecking = False

deprecation_warnings = False

remote_user = penas
private_key_file = ~/.ssh/
penas@penas-workstation-VirtualBox:~/CPE232_penas/Nagios$ cat inventory
server1
CentOS
penas@penas-workstation-VirtualBox:~/CPE232_penas/Nagios$ cat main_playbook.yml
---
#Penas
- hosts: all
  become: true
  pre_tasks:

  - name: Update repository index (CentOS)
    tags: always
    dnf:
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: Install Updates (CentOS)
    tags: always
    dnf:
      update_only: yes
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: Install Updates (Ubuntu)
    tags: always
    apt:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "Ubuntu"
penas@penas-workstation-VirtualBox:~/CPE232_penas/Nagios$
```

Figure 1.5. Creating a new ansible structure where in this figure shows the basic input of the initial main playbook

```
penas@penas-workstation-VirtualBox:~/CPE232_penas/Nagios$ ansible-playbook --ask-become-pass main_playbook.yml
BECOME password:

PLAY [all] *********************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [CentOS]
ok: [server1]

TASK [Update repository index (CentOS)] ****************************************************************************
skipping: [server1]
ok: [CentOS]

TASK [Install Updates (CentOS)] ************************************************************************************
skipping: [server1]
ok: [CentOS]

TASK [Install Updates (Ubuntu)] ************************************************************************************
skipping: [CentOS]
ok: [server1]

PLAY RECAP *********************************************************************************************************
CentOS                     : ok=3    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
server1                    : ok=2    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0

penas@penas-workstation-VirtualBox:~/CPE232_penas/Nagios$
```

Figure 1.6. Shows the output of the initial main playbook which it consist of success Update in Ubuntu and CentOS

4. When initiating roles the admin should create directories using the command **mkdir roles** where we store multiple roles for the main playbook to use. Inside of the **roles** directory, make another directory regarding the types of roles where in this case it'll be named as **nagios4-CentOS** for CentOS Servers and **nagios4-Ubuntu** for Ubuntu Servers

   Lastly, inside the respective directory of the role itself create another directory named as **tasks** which consist a yml file named as **main.yml,** to easily achieve this last step simply input the command **touch nagios4-CentOS/tasks/main.yml** and **touch nagios4-Ubuntu/tasks/main.yml**

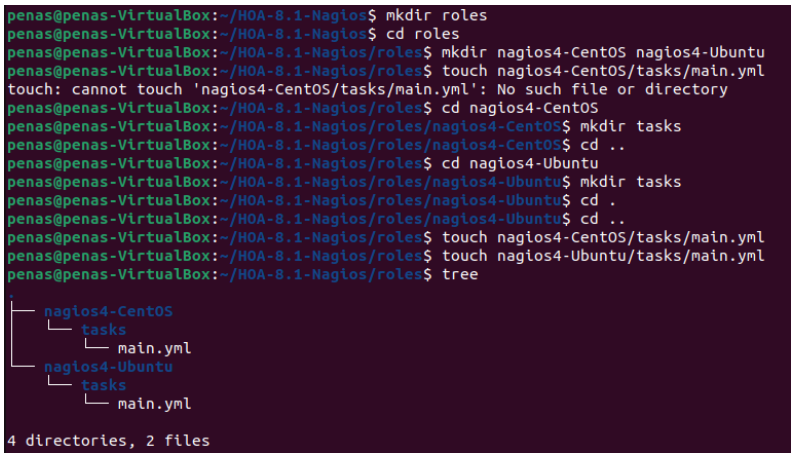


Figure 1.5. Creating Directories for Roles specifically ***nagios4-CentOS*** and ***nagios4-Ubuntu***

5. After creating the subdirectories in Roles Directories, proceed to input the commands on your main.yml of nagios4-Ubuntu

Command Name	***Function***
Installing Nagios4 Dependecies and Libraries	Installs the pre-requisites of the application in order to install the Nagios more efficient
Install passlib Python Package	A password hashing library that will be used when setting up password
Creating a directory for the downloaded files	Creating a new directory in ~nagios where all the downloaded files will be inserted
Downloading and Extracting Nagios4 from Github	Downloads and Extract the Nagios from Github using an external link
Creating Users and group, Compiling and Installation of Nagios4	Creates User and Groups within the Nagios; Compiles all the files from GitHub Link and Installs it

Downloading and Extracting plugins of Nagios4 from Github	Downloading and Extracting Plugins of Nagios4 from GitHub with the destination to ~nagios
Compiling and Installing Nagios4 plugins	Complies and Installs the downloaded Nagios4 plugins from Github
Setting User and Password	Setting the desired User and Password as: *User: ivpenas* *Password: 2010167*
Confirmation of Nagios4 is enabled	Commands that oversees the Nagios4 application is enabled and connected to the server
Confirmation of httpd is enabled	Commands that oversees the httpd is enabled and connected to the server

Table 1.1. Shows brief information on what a set of commands do in the playbook

Ubuntu:

```
#Penas-Ubuntu
- name: Installing Nagios4 Dependecies and Libraries
  tags: dependecies, libraries
  apt:
    name:
      - autoconf
      - gcc
      - libc6
      - make
      - wget
      - unzip
      - apache2
      - php
      - libapache2-mod-php7.4
      - libgd-dev
      - openssl
      - libssl-dev
      - autoconf
      - gcc
      - libc6
      - libmcrypt-dev
      - make
      - libssl-dev
      - wget
      - bc
      - gawk
      - dc
      - build-essential
      - snmp
      - libnet-snmp-perl
      - gettext
      - python3-pip
      - python3
    state: latest

- name: Install passlib Python Package
  pip:
    name: passlib

- name: Creating a directory for the downloaded files
  file:
    path: ~/nagios
    state: directory
```

```
- name: Setting User and Password
  community.general.htpasswd:
    path: /usr/local/nagios/etc/htpasswd.users
    name: ivpenas
    password: 2010167

- name: Confirmation of Nagios4 is enabled
  service:
    name: nagios
    state: restarted
    enabled: true

- name: Confirmation of httpd is enabled
  service:
    name: apache2
    state: restarted
    enabled: true
```

```
- name: Downloading and Extracting Nagios4 from Github
  unarchive:
    src: https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.6.tar.gz
    dest: ~/nagios
    remote_src: yes
    mode: 0755
    owner: root
    group: root

- name: Creating Users and group, Compiling and Installation of Nagios4
  shell: |
    cd ~/nagios/nagioscore-*
    sudo ./configure --with-httpd-conf=/etc/apache2/sites-enabled
    sudo make all
    sudo make install-groups-users
    sudo usermod -a -G nagios www-data
    sudo make install
    sudo make install-daemoninit
    sudo make install-commandmode
    sudo make install-config
    sudo make install-webconf
    sudo a2enmod rewrite
    sudo a2enmod cgi

- name: Downloading and Extracting plugins of Nagios4 from Github
  unarchive:
    src: https://github.com/nagios-plugins/nagios-plugins/archive/release-2.3.3.tar.gz
    dest: ~/nagios
    remote_src: yes
    mode: 0755
    owner: root
    group: root

- name: Compiling and Installing Nagios4 plugins
  shell: |
    cd ~/nagios/nagios-plugins*
    ./tools/setup
    ./configure
    make
    make install
```

Figure 1.6-8. The Playbook of **nagios4-Ubuntu** where it installs the Nagios Application and connects to the server

6. Then create the same set of commands in the playbook of nagios4-CentOS named **main.yml**, there will be minor changes in the playbook of CentOS as some of the commands were different than Ubuntu yet it still serves the same functions.

Command Name	***Function***
Installing Nagios4 Dependecies and Libraries	Installs the pre-requisites of the application in order to install the Nagios more efficient
Install passlib Python Package	A password hashing library that will be used when setting up password
Creating a directory for the downloaded files	Creating a new directory in ~nagios where all the downloaded files will be inserted
Downloading and Extracting Nagios4 from Github	Downloads and Extract the Nagios from Github using an external link
Creating Users and group, Compiling and Installation of Nagios4	Creates User and Groups within the Nagios; Compiles all the files from GitHub Link and Installs it

Downloading and Extracting plugins of Nagios4 from Github	Downloading and Extracting Plugins of Nagios4 from GitHub with the destination to ~nagios
Compiling and Installing Nagios4 plugins	Complies and Installs the downloaded Nagios4 plugins from Github
Setting User and Password	Setting the desired User and Password as: *User: ivpenas* *Password: 2010167*
Confirmation of Nagios4 is enabled	Commands that oversees the Nagios4 application is enabled and connected to the server
Confirmation of httpd is enabled	Commands that oversees the httpd is enabled and connected to the server

Table 1.1. Shows brief information on what a set of commands do in the playbook

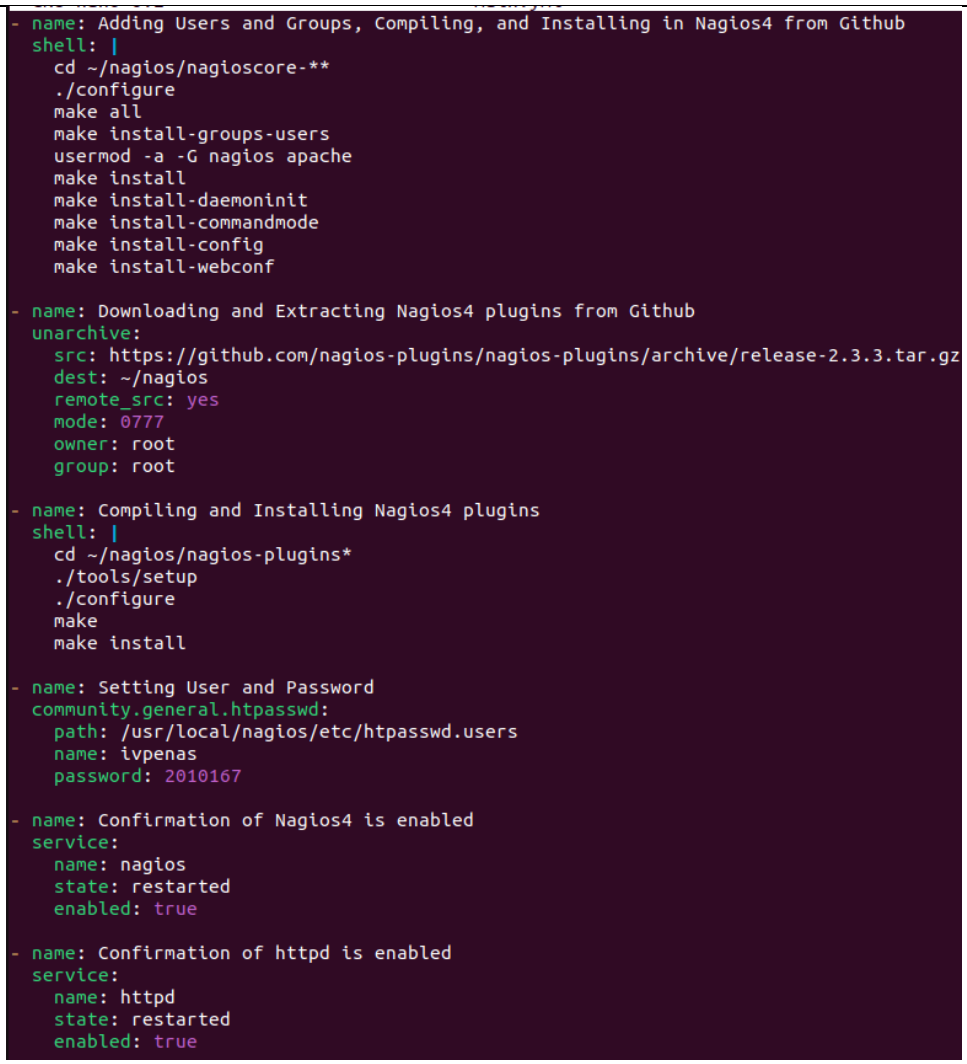
CentOS:

```
#Penas-CentOS
- name: Installing Nagios4 Dependecies and Libraries
  tags: dependecies, libraries
  yum:
    name:
      - gcc
      - glibc
      - glibc-common
      - perl
      - httpd
      - php
      - wget
      - gd
      - gd-devel
      - openssl-devel
      - gcc
      - glibc
      - glibc-common
      - make
      - gettext
      - automake
      - autoconf
      - wget
      - openssl-devel
      - net-snmp
      - net-snmp-utils
      - python2-pip
    state: latest

- name: Install passlib Python Package
  pip:
    name: passlib

- name: Creating a directory for the downloaded files
  file:
    path: ~/nagios
    state: directory

- name: Downloading and Extracting Nagios4 from Github
  unarchive:
    src: https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.6.tar.gz
    dest: ~/nagios
    remote_src: yes
    mode: 0777
```



```
- name: Adding Users and Groups, Compiling, and Installing in Nagios4 from Github
  shell: |
    cd ~/nagios/nagioscore-**
    ./configure
    make all
    make install-groups-users
    usermod -a -G nagios apache
    make install
    make install-daemoninit
    make install-commandmode
    make install-config
    make install-webconf

- name: Downloading and Extracting Nagios4 plugins from Github
  unarchive:
    src: https://github.com/nagios-plugins/nagios-plugins/archive/release-2.3.3.tar.gz
    dest: ~/nagios
    remote_src: yes
    mode: 0777
    owner: root
    group: root

- name: Compiling and Installing Nagios4 plugins
  shell: |
    cd ~/nagios/nagios-plugins*
    ./tools/setup
    ./configure
    make
    make install

- name: Setting User and Password
  community.general.htpasswd:
    path: /usr/local/nagios/etc/htpasswd.users
    name: ivpenas
    password: 2010167

- name: Confirmation of Nagios4 is enabled
  service:
    name: nagios
    state: restarted
    enabled: true

- name: Confirmation of httpd is enabled
  service:
    name: httpd
    state: restarted
    enabled: true
```

Figure 1.9-10. The Playbook of **nagios4-CentOS** where it installs the Nagios Application and connects to the server

7. Dont you forget your initial global_playbook.yml, since we've initiated roles between **nagios4-CentOS** and **nagios4-Ubuntu,** the admin should edit and append some commands from the global_playbook.yml. Make sure you are in the same directory of the playbook.

```yaml
---

- hosts: all
  become: true
  pre_tasks:

  - name: Update and upgrade remote in Ubuntu servers
    apt:
      update_cache: yes
      upgrade: 'yes'
    when: ansible_distribution == "Ubuntu"

  - name: Installing dnf and epel-release
    yum:
      name:
        - epel-release
        - dnf
    when: ansible_distribution == "CentOS"

  - name: Update and upgrade remote CentOS server
    dnf:
      update_cache: yes
      name:  "*"
      state: latest
    when: ansible_distribution == "CentOS"

  - name: Dpkg fixing in Ubuntu Servers
    shell: |
      dpkg --configure -a
    when: ansible_distribution == "Ubuntu"

- hosts: nagios4-Ubuntu
  become: true
  roles:
    - nagios4-Ubuntu

- hosts: nagios4-CentOS
  become: true
  roles:
    - nagios4-CentOS
```

Figure 1.11. The Playbook of **global_playbook.yml** where initiates the execution of both roles and other separate commands such as Updating and Upgrading CentOS and Ubuntu Servers; Installation of DNF and Epel Release; and Configuring Dpkg in Ubuntu Servers as a partial requirement of Nagios

8. When the playbooks are correctly configured, run the command **ansible-playbook --ask-become-pass global_playbook.yml** which executes the main playbook of this folder.

Outputs:

```
ok: [server_1]
ok: [Cent-OS]

TASK [Update and upgrade remote in Ubuntu servers] ************************************
skipping: [Cent-OS]
ok: [server_1]

TASK [Installing dnf and epel-release] ************************************************
skipping: [server_1]
ok: [Cent-OS]

TASK [Update and upgrade remote CentOS server] ****************************************
skipping: [server_1]
ok: [Cent-OS]

TASK [Dpkg fixing in Ubuntu Servers] **************************************************
skipping: [Cent-OS]
changed: [server_1]

PLAY [nagios4-CentOS] *****************************************************************

TASK [Gathering Facts] ****************************************************************
ok: [Cent-OS]

TASK [nagios4-CentOS : Installing Nagios4 Dependecies and Libraries] ******************
ok: [Cent-OS]

TASK [nagios4-CentOS : Install passlib Python Package] ********************************
ok: [Cent-OS]

TASK [nagios4-CentOS : Creating a directory for the downloaded files] *****************
ok: [Cent-OS]

TASK [nagios4-CentOS : Downloading and Extracting Nagios4 from Github] ****************
ok: [Cent-OS]

TASK [nagios4-CentOS : Adding Users and Groups, Compiling, and Installing in Nagios4 from
Github] ***
changed: [Cent-OS]

TASK [nagios4-CentOS : Downloading and Extracting Nagios4 plugins from Github] *********
ok: [Cent-OS]

TASK [nagios4-CentOS : Compiling and Installing Nagios4 plugins] **********************
changed: [Cent-OS]

TASK [nagios4-CentOS : Setting User and Password] *************************************
ok: [Cent-OS]

TASK [nagios4-CentOS : Confirmation of Nagios4 is enabled] ****************************
changed: [Cent-OS]

TASK [nagios4-CentOS : Confirmation of httpd is enabled] ******************************
changed: [Cent-OS]
```

```
PLAY [nagios4-Ubuntu] ***********************************************************

TASK [Gathering Facts] ***********************************************************
ok: [server_1]

TASK [nagios4-Ubuntu : Installing Nagios4 Dependecies and Libraries] *******************
ok: [server_1]

TASK [nagios4-Ubuntu : Install passlib Python Package] *********************************
ok: [server_1]

TASK [nagios4-Ubuntu : Creating a directory for the downloaded files] ******************
ok: [server_1]

TASK [nagios4-Ubuntu : Downloading and Extracting Nagios4 from Github] *****************
ok: [server_1]

TASK [nagios4-Ubuntu : Creating Users and group, Compiling and Installation of Nagios4] *
**
changed: [server_1]

TASK [nagios4-Ubuntu : Downloading and Extracting plugins of Nagios4 from Github] ******
ok: [server_1]

TASK [nagios4-Ubuntu : Compiling and Installing Nagios4 plugins] ***********************
changed: [server_1]

TASK [nagios4-Ubuntu : Setting User and Password] ***************************************
ok: [server_1]

TASK [nagios4-Ubuntu : Confirmation of Nagios4 is enabled] *****************************
changed: [server_1]

TASK [nagios4-Ubuntu : Confirmation of httpd is enabled] *******************************
changed: [server_1]

PLAY RECAP ***********************************************************************
Cent-OS                    : ok=14   changed=4    unreachable=0    failed=0    skipped=2
   rescued=0    ignored=0
server_1                   : ok=14   changed=5    unreachable=0    failed=0    skipped=2
   rescued=0    ignored=0
```

Figure 1.12-14. Shows the Output of the PLAY RUN after executing the main playbook which is **global_playbook.yml**

9. To Verify if the Nagios was fully installed within the Servers, Open a Brower and insert the respective IP Adress of the server along with **/nagios,** In Server 1 - 192.168.56.112/nagios while in CentOS - 192.168.56.111/nagios

Verification:

Ubuntu:

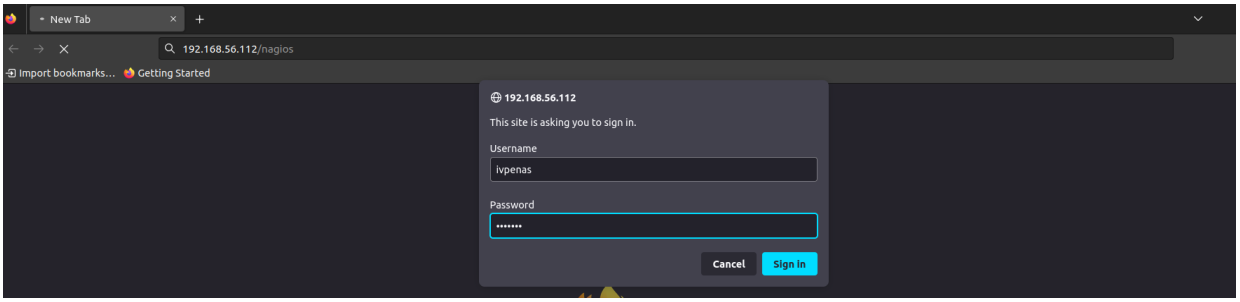


Figure 1.15. Shows a user verification where the admin will input the correct Username and Password

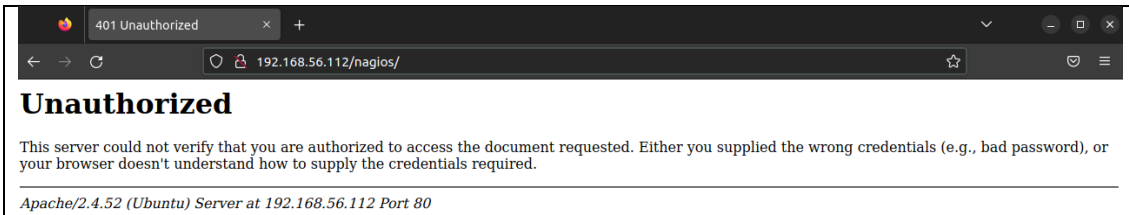


Figure 1.16. If the Credential was not correct it'll show the Unauthorized page

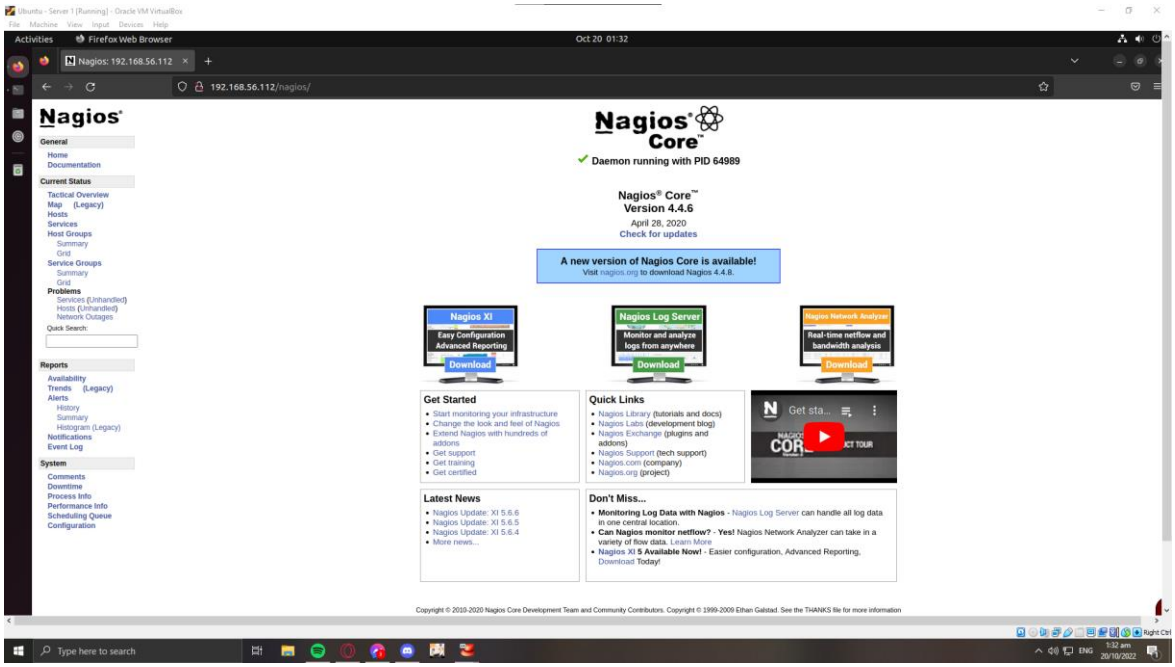


Figure 1.17. If the Credential was correct it'll show the Nagios page

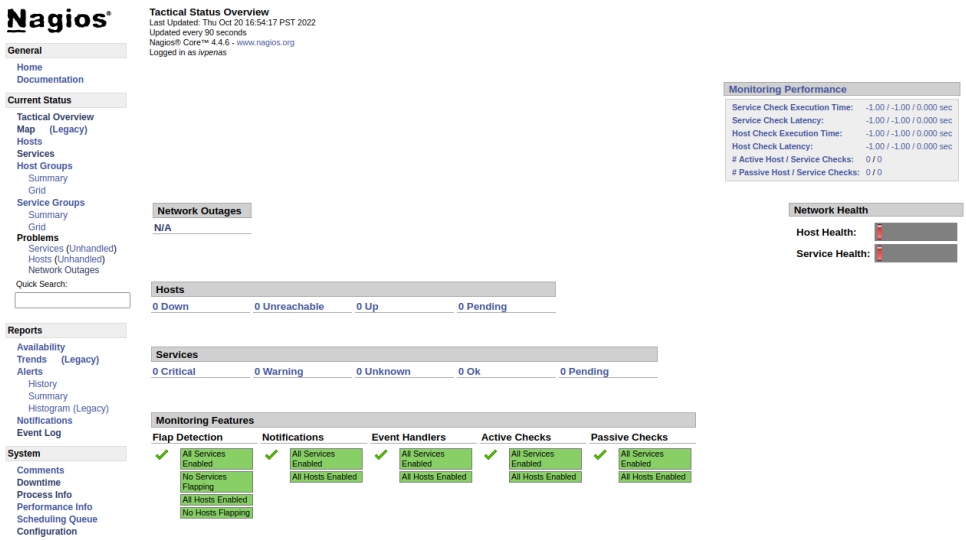


Figure 1.18. Shows the Tactical Overview of the Server

CentOS:

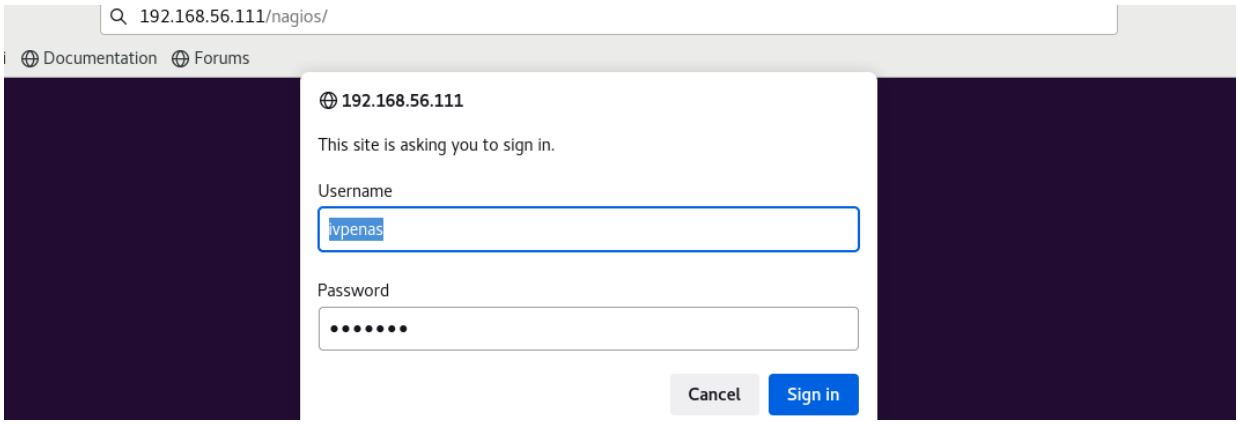


Figure 1.19. Shows a user verification where the admin will input the correct Username and Password

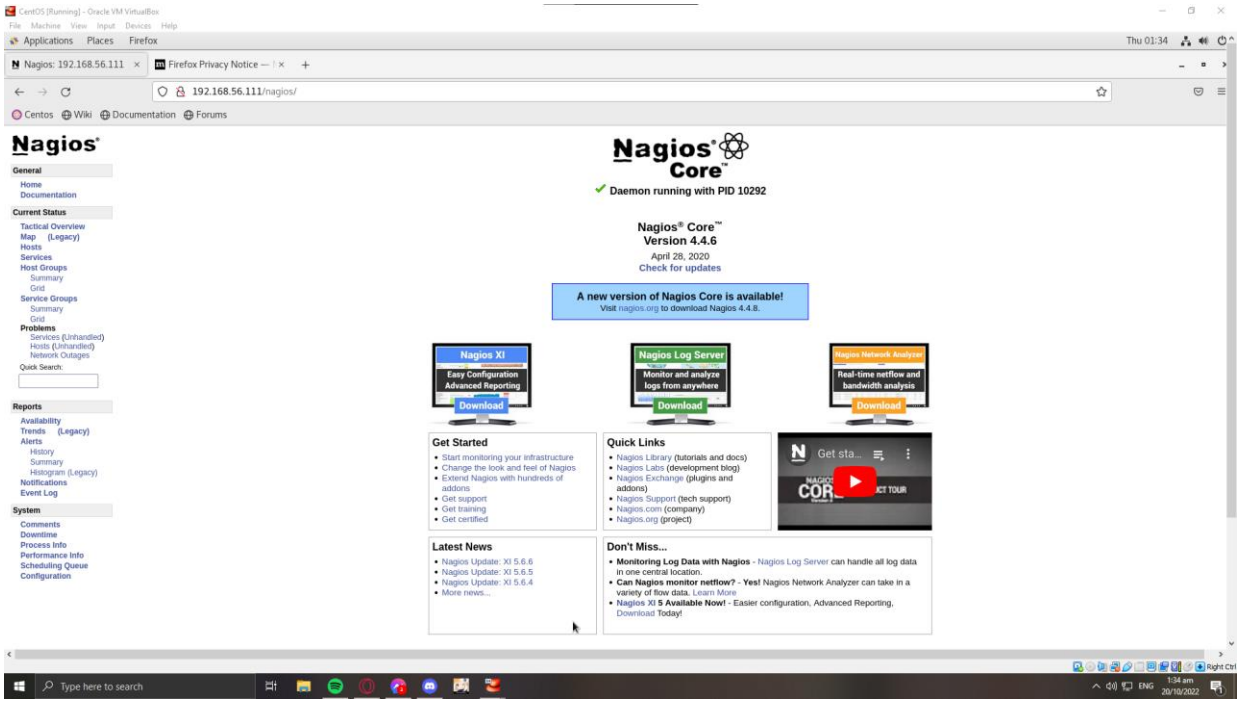


Figure 1.20. If the Credential was correct it'll show the Nagios page

Figure 1.21. Shows the Downtime in CentOS Server

10. When the Verification was done, fully sync the cloned folder to Github using the command '**git add ***' which adds a changed directory, '**git commit -m "ANYMESSG"**' Commits the Changes are made within the repository folder, and '**git push**' to sync and save the folder to the Git Repository.

```
penas@penas-VirtualBox:~/HOA-8.1-Nagios$ git add *
penas@penas-VirtualBox:~/HOA-8.1-Nagios$ git commit -m "Last Commit"
[main 43e909f] Last Commit
 4 files changed, 229 insertions(+), 16 deletions(-)
 create mode 100644 roles/nagios4-CentOS/tasks/main.yml
 create mode 100644 roles/nagios4-Ubuntu/tasks/main.yml
penas@penas-VirtualBox:~/HOA-8.1-Nagios$ git push
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (11/11), 2.10 KiB | 2.10 MiB/s, done.
Total 11 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
```

Figure 1.20. Insert the commands to update the Repository

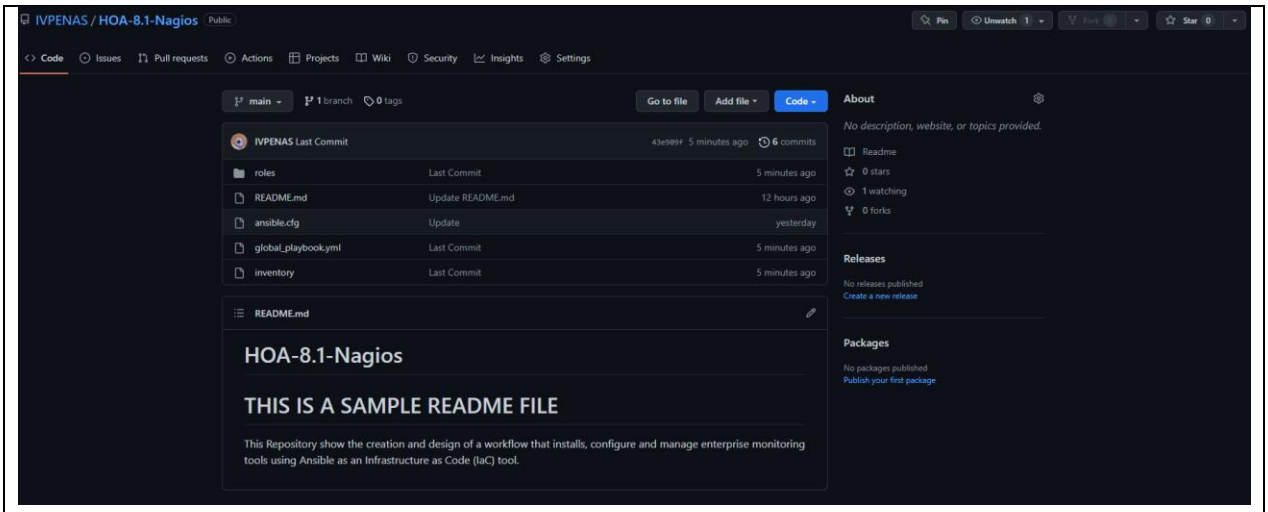


Figure 1.21. The Repository is up to date

GitHub Repository Link: *https://github.com/IVPENAS/HOA-8.1-Nagios.git*

Reflections:

Answer the following:

1. What are the benefits of having an availability monitoring tool?

   Availability monitoring tools can be easily identified as our Task Manager in our Windows OS computers, which illustrates the *servers* and *applications opened* and their *activities* using **tables and graphs to identify their statuses**. These kinds of monitory tools are mostly used in businesses specifically companies that handle multiple servers and databases which the System Administrators handle in order to *prevent situations like DDoS or Bugs within the Servers* by notifying them beforehand before it can make catastrophic damage to the business where it can affect the structure of the Business and also the revenue.

   The Nagios Application is one example of an Availability Monitoring Tool, where its services include a [1] **Comprehensive Monitoring** of the server, applications, network protocols, and Operating systems. [2] **Problem Remediation** which allows the System Administrator to be notified whenever a problem or issues occur within the system and automatically restarts any failed applications or services. [3] **UPS Backup system**, and more, which can benefit the networks.

Conclusions:

The Nagios Application is one of the monitoring tool open-source applications that a System Administrator uses, whereas it serves as the 'Task Manager' that monitors multiple types of devices such as simple Computer Systems, Servers, Network Protocols, or Operating Systems which can benefit companies that handles multiple Network and Servers as it has features that secures the data, structure and security

infrastructures. Both CentOS and Ubuntu servers were able to install and connect to the Nagios Application by verifying it on the browser, inputting the IP Address of the respective server along with the path ***'/nagios',*** and inputting the correct credentials. The installation took some time as the student encountered multiple errors, of both Ubuntu and CentOS being unable to update and using their old update file which resorted to multiple reinstallations within the system and repeating the ssh connectivity. In the end, the student was able to attain multiple objectives in this Hand-On-Activity whereas creating a custom ansible playbook by installing availability monitoring tools specifically the Nagios application, and initiating roles of both Ubuntu and CentOS Servers that provides efficiency for the System Administrator to download, extract, install, update, and upgrade multiple Servers in one go.

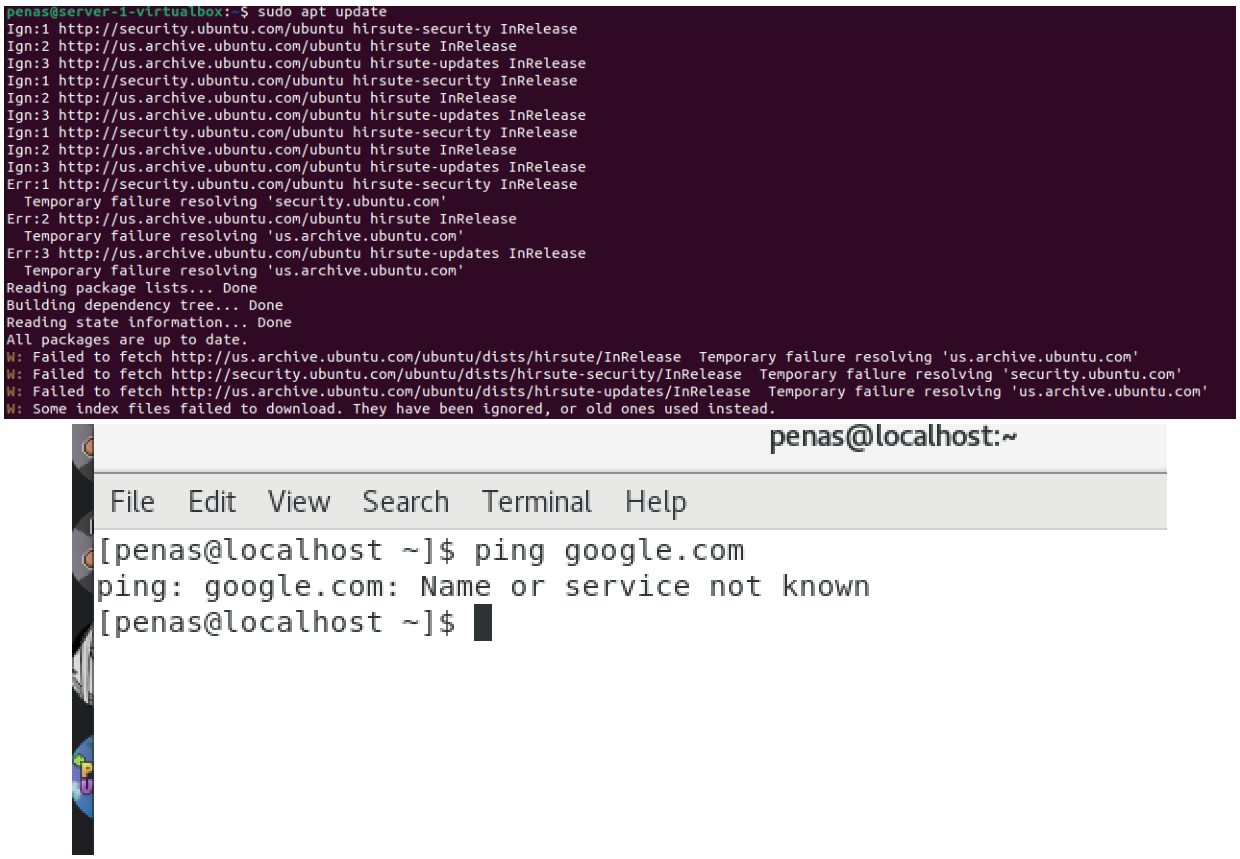


Figure 2.1. Shows one of the error that the student encountered during the activity