



Laboratorio 8: Usamos Ansible para automatizar la instalación de un servidor web.

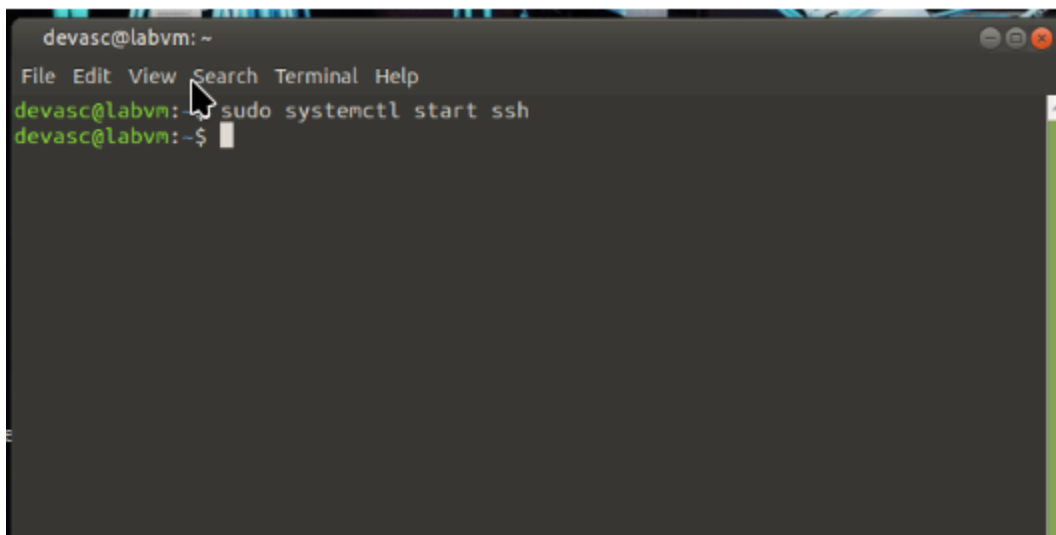
Sergio Sebastian Pezo Jimenez - 20224087G

En este laboratorio documentaremos el proceso de automatización de la instalación y configuración de un servidor web utilizando Ansible en nuestra máquina virtual DEVASC, verificaremos la conectividad con el servidor web, y crearemos varios playbooks para automatizar la instalación y configuración de un servidor Apache.

Iniciamos nuestra máquina virtual DEVASC

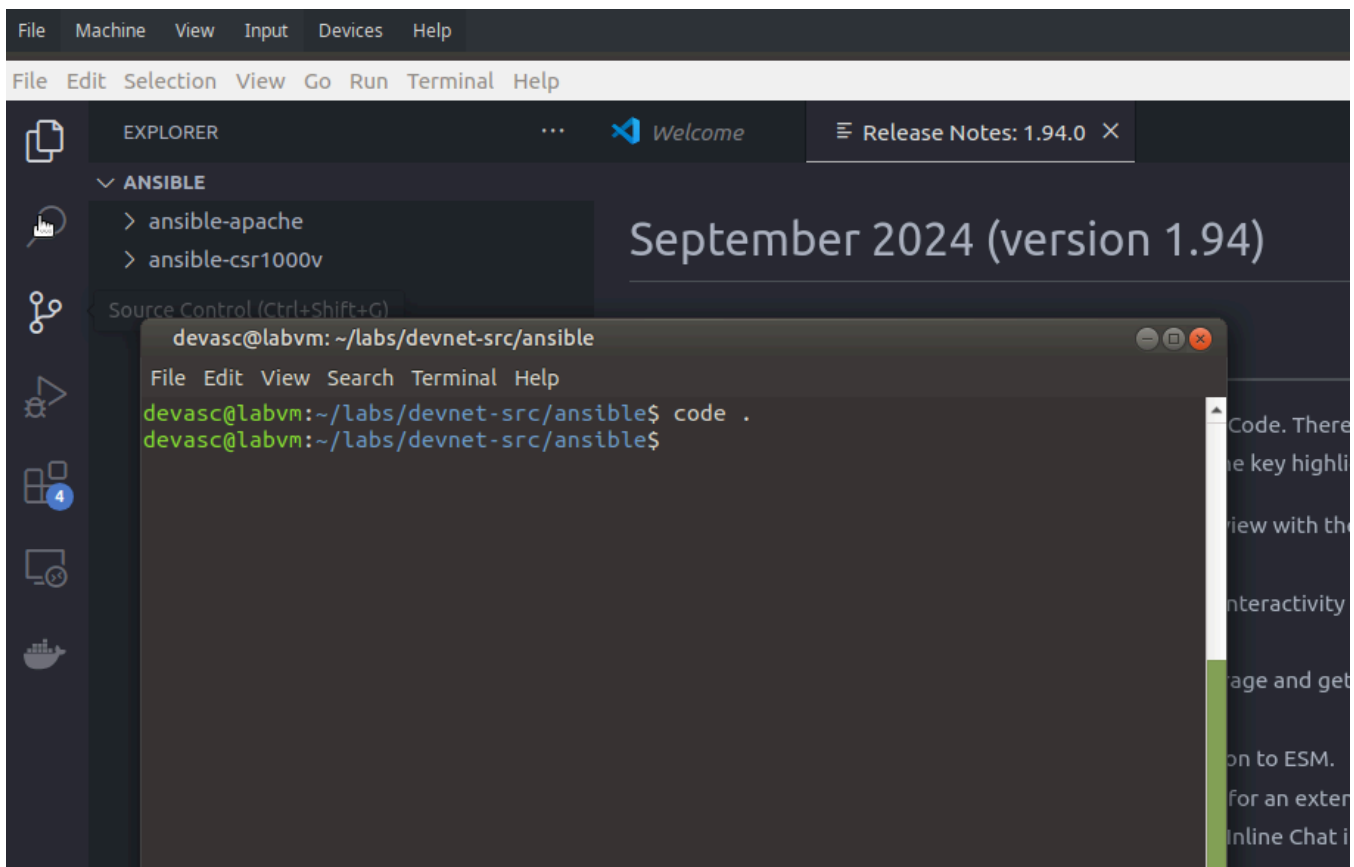
Empecemos configurando Ansible

Abrimos el terminal de nuestra máquina virtual y habilitamos el servidor `SSH` el cual viene previamente instalado en nuestro dispositivo, por lo que ya no los instalamos.

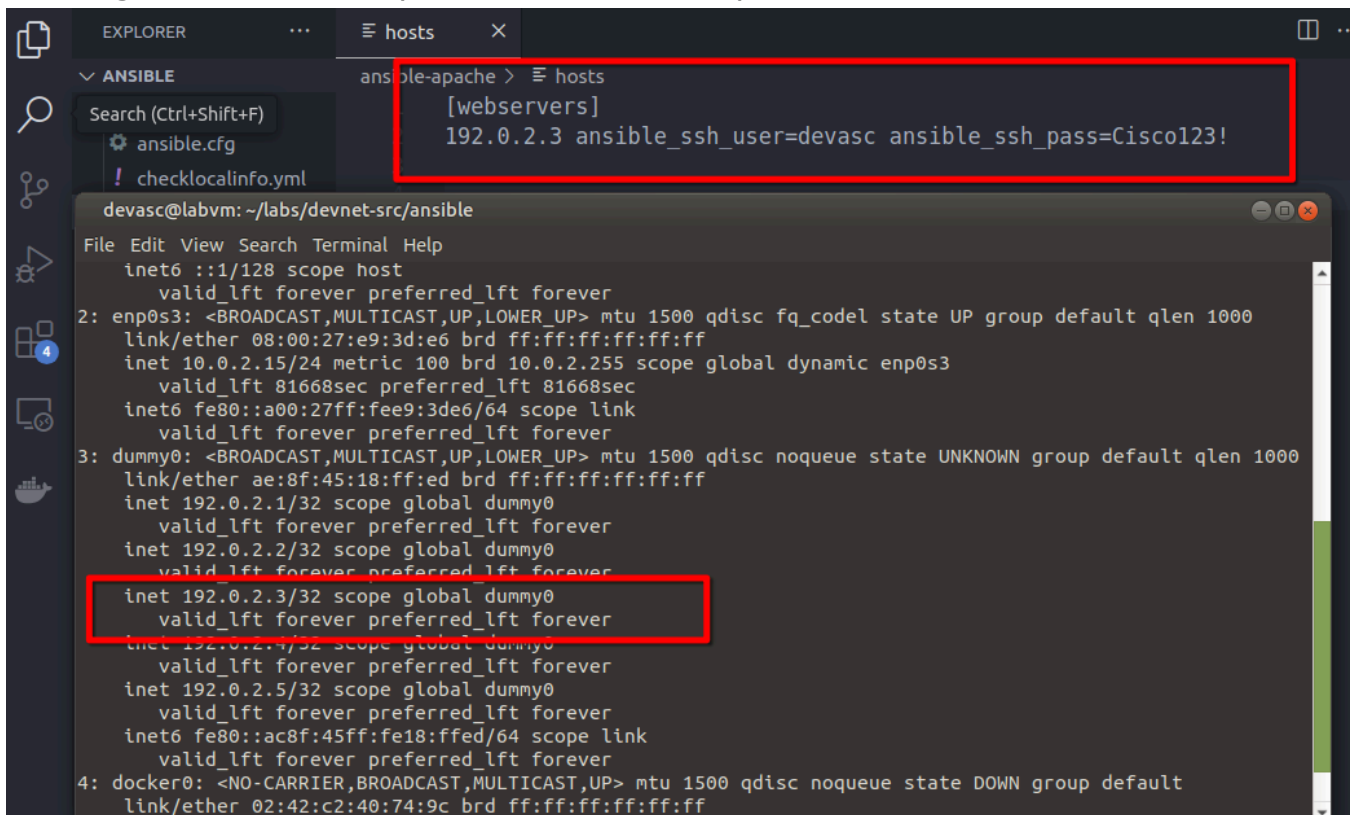


```
devasc@labvm: ~  
File Edit View Search Terminal Help  
devasc@labvm:~$ sudo systemctl start ssh  
devasc@labvm:~$
```

Abrimos VS Code en `/labs/devnet-src/ansible`



Y configuramos los host para Ansible con un ip estática nuestra



Editamos el `ansible.cfg` para especificar a Ansible donde encontrar a el archivo de inventario.

Verifiquemos comunicación con el servidor local.

Usamos el módulo `ping` para verificar que Ansible puede conectarse al servidor web.

```
devasc@labvm:~/labs/devnet-src/ansible$ cd ansible-apache/
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible webservers -m ping

192.0.2.3 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

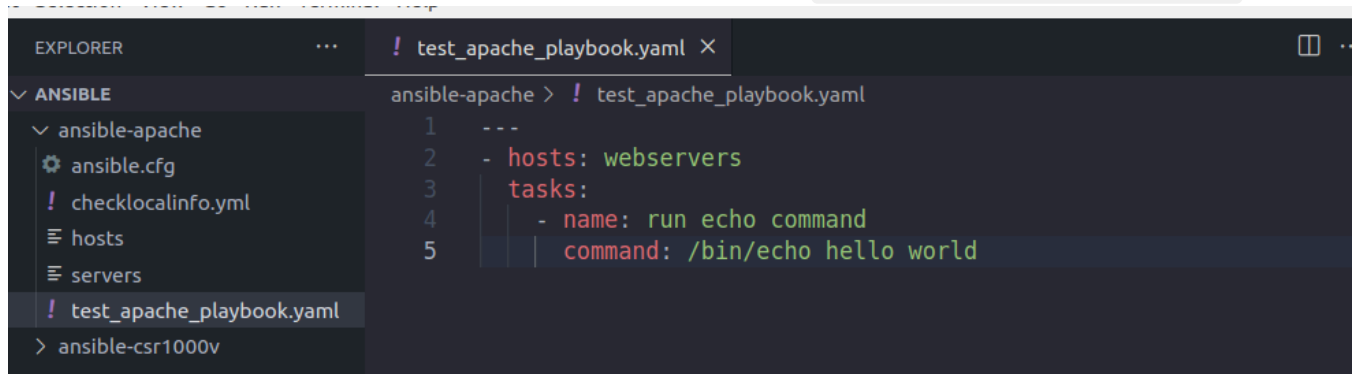
Ahora verificamos si puede comunicarse con este.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible webservers -m command -a "/bin/echo hello world"
192.0.2.3 | CHANGED | rc=0 >>
hello world
```

Todo bien.

Creamos un Ansible Playbook para automatizar la instalación del servidor web

Crearemos nuestro primer Playbook en el archivo `test_apache_playbook.yaml`.



```
EXPLORER  ...  ! test_apache_playbook.yaml X

ANSIBLE
└─ ansible-apache
   ├── ansible.cfg
   ├── ! checklocalinfo.yml
   ├── hosts
   ├── servers
   └── ! test_apache_playbook.yaml
       > ansible-csr1000v

ansible-apache > ! test_apache_playbook.yaml
1  ---
2  - hosts: webservers
3    tasks:
4      - name: run echo command
5        command: /bin/echo hello world
```

Ejecutamos nuestro playbook para probar nuestro server group.

```

devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible-playbook -v test_apache_playbook.yaml
Using /home/devasc/labs/devnet-src/ansible/ansible-apache/ansible.cfg as config file

PLAY [webservers] *****

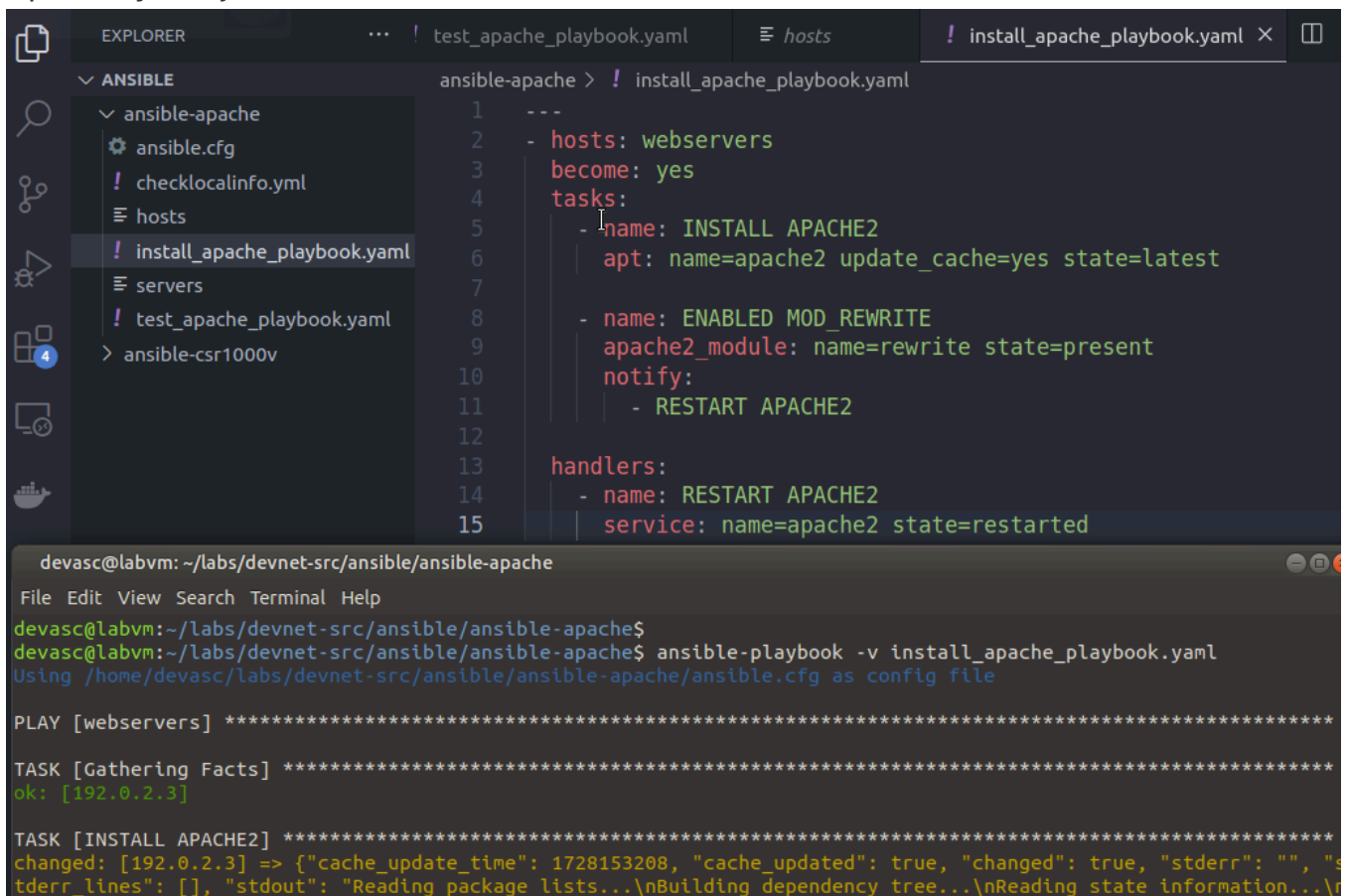
TASK [Gathering Facts] *****
ok: [192.0.2.3]

TASK [run echo command] *****
changed: [192.0.2.3] => {"changed": true, "cmd": ["/bin/echo", "hello", "world"], "delta": "0:00:00.001492", "en
": "2024-10-05 18:21:16.028152", "rc": 0, "start": "2024-10-05 18:21:16.026660", "stderr": "", "stderr_lines": [
, "stdout": "hello world", "stdout_lines": ["hello world"]}

PLAY RECAP *****
192.0.2.3          : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=
0

```

Ahora, que sabemos que todo va bien, creamos un nuevo playbook para instalar Apache y lo ejecutamos.



The screenshot shows a VS Code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project structure for 'ansible-apache' with files like 'ansible.cfg', 'checklocalinfo.yml', 'hosts', 'install_apache_playbook.yaml', 'servers', 'test_apache_playbook.yaml', and 'ansible-csr1000v'. The 'install_apache_playbook.yaml' file is open in the editor, showing the following content:

```

1  ---
2  - hosts: webservers
3    become: yes
4    tasks:
5      - name: INSTALL APACHE2
6        apt: name=apache2 update_cache=yes state=latest
7
8      - name: ENABLED MOD_REWRITE
9        apache2_module: name=rewrite state=present
10       notify:
11         - RESTART APACHE2
12
13     handlers:
14       - name: RESTART APACHE2
15         service: name=apache2 state=restarted

```

The terminal at the bottom shows the execution of the 'install_apache_playbook.yaml' playbook. It starts with the same 'PLAY [webservers]' and 'TASK [Gathering Facts]' as the previous output. The next task is 'TASK [INSTALL APACHE2]', which is successful and shows the following output:

```

changed: [192.0.2.3] => {"cache_update_time": 1728153208, "cache_updated": true, "changed": true, "stderr": "", "s
tderr_lines": [], "stdout": "Reading package lists...\nBuilding dependency tree...\nReading state information...\n

```

Todo ok.

```
PLAY RECAP *****
192.0.2.3 : ok=4 changed=3 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2024-10-05 18:33:40 UTC; 1min 21s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 13949 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 13962 (apache2)
    Tasks: 55 (limit: 4601)
   Memory: 5.1M
    CGroup: /system.slice/apache2.service
            └─13962 /usr/sbin/apache2 -k start
              13963 /usr/sbin/apache2 -k start
              13964 /usr/sbin/apache2 -k start

Oct 05 18:33:40 labvm systemd[1]: Starting The Apache HTTP Server...
Oct 05 18:33:40 labvm systemd[1]: Started The Apache HTTP Server.
```

Agreguemos opciones a nuestro playbook

Creamos un nuevo playbook `install_apache_options_playbook.yaml` y agregamos las opciones para activar el servidor en el puerto 8081.

```

e Edit Selection View Go Run Terminal Help
:he_playbook.yaml  hosts  ! install_apache_playbook.yaml  ! install_apache_options_playbook.yaml  ×  ...

ansible-apache > ! install_apache_options_playbook.yaml
1  ---
2  - hosts: webservers
3    become: yes
4    tasks:
5      - name: INSTALL APACHE2
6        apt: name=apache2 update_cache=yes state=latest
7
8      - name: ENABLED MOD_REWRITE
9        apache2_module: name=rewrite state=present
10       notify:
11         - RESTART APACHE2
12
13       - name: APACHE2 LISTEN ON PORT 8081
14         lineinfile: dest=/etc/apache2/ports.conf regexp="^Listen 80" line="Listen 8081" state=p
15         notify:
16           - RESTART APACHE2
17
18       - name: APACHE2 VIRTUALHOST ON PORT 8081
19         lineinfile: dest=/etc/apache2/sites-available/000-default.conf regexp="^<VirtualHost \*"
20         notify:
21           - RESTART APACHE2
22
23     handlers:
24       - name: RESTART APACHE2
25         service: name=apache2 state=restarted

```

Examinamos los dos archivos que se modificarán:

```
c-05 10.55.10 lab01n1 system@1: started the apache http server.
```

```

yasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ cat /etc/apache2/ports.conf
#
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

```

sten 80

```
fModule ssl_module>  
    Listen 443  
IfModule>
```

```
fModule mod_gnutls.c>  
    Listen 443  
IfModule>
```

```
vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

```
vasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ cat /etc/apache2/sites-available/000-default.conf
VirtualHost *:80>
```

```
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com
```

```
ServerAdmin webmaster@localhost
DocumentRoot /var/www/html
```

```
# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn
```

```
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
```

```
# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
```

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible-playbook install apache options playbook.yaml
```

```
PLAY [webservers] *****
```

```
TASK [Gathering Facts] *****
ok: [192.0.2.3]
```

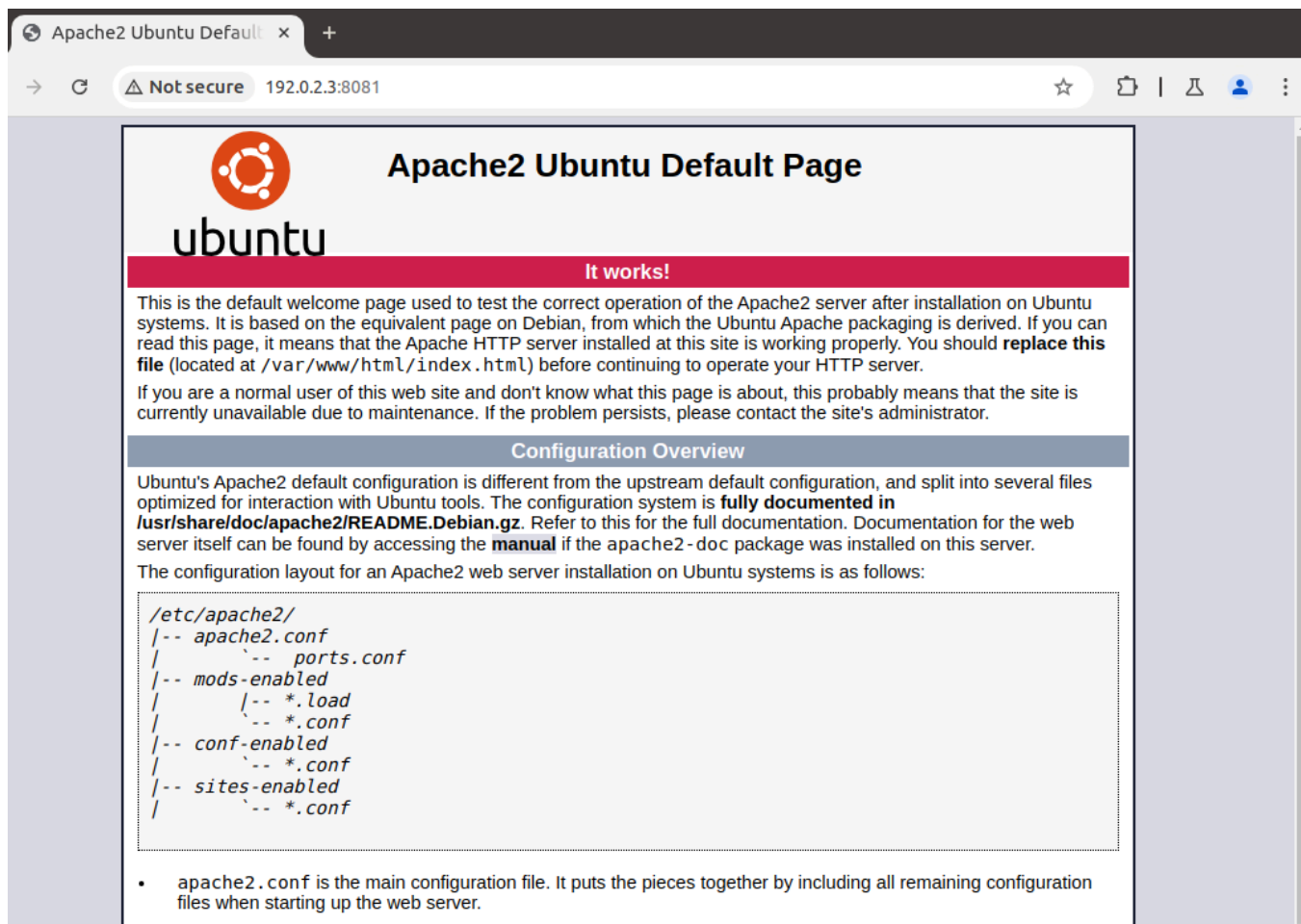
```
TASK [INSTALL APACHE2] *****
ok: [192.0.2.3]
```

```
TASK [ENABLED MOD_REWRITE] *****
ok: [192.0.2.3]
```

```
TASK [APACHE2 LISTEN ON PORT 8081] *****
changed: [192.0.2.3]
```

```
TASK [APACHE2 VIRTUALHOST ON PORT 8081] *****
changed: [192.0.2.3]
```

¡Y funcionó!



Conclusiones

1. **Configuración de Ansible:** Aprendimos a configurar Ansible en una máquina virtual DEVASC, incluyendo la habilitación del servidor SSH y la configuración de los hosts y el archivo `ansible.cfg`.
2. **Verificación de Conexiones:** Utilizamos el módulo `ping` de Ansible para verificar la conectividad con el servidor web local, asegurando que Ansible puede comunicarse correctamente con el servidor.
3. **Automatización con Playbooks:** Creamos y ejecutamos varios playbooks de Ansible para automatizar la instalación y configuración de un servidor web Apache, demostrando la capacidad de Ansible para gestionar configuraciones de servidores de manera eficiente.
4. **Personalización de Playbooks:** Añadimos opciones a nuestros playbooks para personalizar la configuración del servidor web, como la activación del servidor en un puerto específico, mostrando la flexibilidad y el poder de Ansible para adaptarse a diferentes necesidades.

5. **Resultados Exitosos:** Todos los pasos fueron ejecutados con éxito, desde la configuración inicial hasta la personalización avanzada, validando la efectividad de Ansible como herramienta de automatización.