

Profesora: Emma Di Battista

Estudiante: Amy Meneses

Carnet: 17-10381

Práctica 3: Docker

Para la ejecución de esta práctica primero se modificó el script de Python “script.py”, en este se agrega un temporizador para realizar la consulta y actualizar el archivo “Inventario.csv” cada 5min. Para esto se creó una función Repeticion() en el que se utilizó la función sleep() de la librería time de Python, esto permite suspender la ejecución del script por un número de segundos, en este caso correspondientes a 5min. Y con el ciclo While(True) se realiza la consulta cada 5min, como se muestra en el siguiente fragmento del script:

```
def Repeticion (): #Consulta cada 5min
    time.sleep(5*60)
    print("El archivo Inventario.csv se actualizo")

while(True): #Se realiza la consulta cada 5min y se actualiza el archivo "Inventario.csv"
    org=getOrganizations()
    getOrganizationsDevices (org)
    Repeticion()
```

Figura 1: Fragmento de “script.py”

Luego, se instaló la imagen de alpine con el siguiente comando:

docker pull alpine

Y se escribió el archivo Dockerfile, que contiene los comandos necesarios para crear la imagen requerida para el funcionamiento de “script.py”. En este se indica la versión de Python y la versión de la imagen Alpine a utilizar, la dirección del directorio de trabajo, en este caso, /DeLab. También, se copia el archivo de requirements.txt con las librerías necesarias para correr el script, y, se instalan con el comando RUN, finalmente, se ejecuta el script con el comando **python script.py**. El Dockerfile creado se muestra en la Figura 2.

```

> SDN_DeLab > Dockerfile
1 FROM python:3.11.0b5-alpine3.16
2 WORKDIR /DeLab
3 COPY requirements.txt ./
4 RUN pip install --no-cache-dir -r requirements.txt
5 COPY . .
6 CMD ["python","script.py"]

```

Figura 2: Dockerfile

También se reescribió el archivo README.md, se añadió la modificación de “script.py” y se indicó la integración del Dockerfile. Además, se modificó el archivo requirements.txt dejando solo las librerías indispensables para que pudiera ejecutarse el script. Se agregó al Stage y finalmente, se hizo el commit de estas modificaciones y el Dockerfile.

```

anym@anym:~/SDN_DeLab$ git status
En la rama master
Tu rama está actualizada con 'origin/master'.

Cambios no rastreados para el commit:
(usa "git add <archivo>..." para actualizar lo que será confirmado)
(usa "git checkout -- <archivo>..." para descartar los cambios en el directorio de trabajo)

    modificado:   README.md
    modificado:   requirements.txt
    modificado:   script.py

Archivos sin seguimiento:
(usa "git add <archivo>..." para incluirlo a lo que se será confirmado)

    Dockerfile

sin cambios agregados al commit (usa "git add" y/o "git commit -a")
anym@anym:~/SDN_DeLab$ git add --all
anym@anym:~/SDN_DeLab$ git status
En la rama master
Tu rama está actualizada con 'origin/master'.

Cambios a ser confirmados:
(usa "git reset HEAD <archivo>..." para sacar del área de stage)

    nuevo archivo: Dockerfile
    modificado:   README.md
    modificado:   requirements.txt
    modificado:   script.py

anym@anym:~/SDN_DeLab$ git commit
[master 7dbecae] Se actualiza el script.py con un temporizador de 5min, para realizar la consulta y actualizar el listado Inventario.py cada 5
min
 4 files changed, 43 insertions(+), 57 deletions(-)
 create mode 100644 Dockerfile
 rewrite requirements.txt (96%)

```

Figura 3: Commit de las modificaciones y el archivo Dockerfile

Se procedió a crear la imagen con Docker, para esto se utilizó el comando docker build ., con este comando se especifica que el Dockerfile está en el directorio local, esto se muestra en la Figura 4. Como esto crea una imagen con un nombre aleatorio, se procedió a ejecutar el comando docker build -t delab . para que la imagen tuviera el nombre delab, como se muestra en la Figura 5.

```

anym@anym:~/SDN_DeLab$ sudo docker build .
[sudo] contraseña para anym:
Sending build context to Docker daemon 102.4kB
Step 1/6 : FROM python:3.11.0b5-alpine3.16
--> 88b6a63e2b12
Step 2/6 : WORKDIR /DeLab
--> Using cache
--> 54fc01972b5a
Step 3/6 : COPY requirements.txt ./
--> f4e6430d7bba
Step 4/6 : RUN pip install --no-cache-dir -r requirements.txt
--> Running in d1251f4cbbec

```

Figura 4: Build de la imagen a partir de Dockerfile en directorio local

```

anym@anym:~/SDN_DeLab$ sudo docker build -t delab .
Sending build context to Docker daemon 102.4kB
Step 1/6 : FROM python:3.11.0b5-alpine3.16
--> 88b6a63e2b12
Step 2/6 : WORKDIR /DeLab
--> Using cache
--> 54fc01972b5a
Step 3/6 : COPY requirements.txt ./
--> Using cache
--> f4e6430d7bba
Step 4/6 : RUN pip install --no-cache-dir -r requirements.txt
--> Using cache
--> c4a22ba8f123
Step 5/6 : COPY . .
--> Using cache
--> bdc65363baa4
Step 6/6 : CMD ["python","script.py"]
--> Using cache
--> 4f8b4933d7ea
Successfully built 4f8b4933d7ea
Successfully tagged delab:latest

```

Figura 5: Build con nombre delab de la imagen a partir de Dockerfile en directorio local

Se verificó que se haya creado la imagen con el comando **docker images**

```

anym@anym:~/SDN_DeLab$ sudo docker images

```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
delab	latest	4f8b4933d7ea	16 minutes ago	69.4MB

Una vez creada la imagen, se ejecutó el comando **docker run -it --name sdn delab**. Con este comando se ejecuta el contenedor deseado, en este caso, se utiliza la opción **-it** para tener los procesos interactivos, la opción **--name** para identificar al contenedor con un nombre y finalmente, la imagen a utilizar, en este caso, **delab**. Una vez ejecutado el comando, se observa cómo se ejecuta el script “script.py” del contenedor, como se muestra en la Figura 6:

```

anym@anym:~/SDN_DeLab$ sudo docker run -it --name sdn delab
La lista de las organizaciones es:

[{'api': {'enabled': True},
  'cloud': {'region': {'name': 'North America'}},
  'id': '573083052582915028',
  'licensing': {'model': 'co-term'},
  'name': 'Next Meraki Org',
  'url': 'https://n18.meraki.com/o/PoiDucs/manage/organization/overview'},

```

Figura 6: Comando run interactivo para ejecutar contenedor de imagen delab.

Pero como el contenedor se está ejecutando en primer plano, por lo tanto, para ejecutarlo en segundo plano, se agrega la opción `-d`, que indica detached. Para verificar que se está ejecutando en segundo plano se utiliza el comando **docker ps** que muestra los contenedores en ejecución. Luego, se utilizó el comando **docker exec** que permite correr un comando en un contenedor en ejecución, en este caso, se coloca `/bin/sh` y con `ls` se listan los archivos del directorio de trabajo del contenedor, en el que se verifica que está `Inventario.csv`.

```

anym@anym:~/SDN_DeLab$ sudo docker run -it -d --name sdn delab
1115a45eaaae03eb1da748ab35c3a851dc81ab46e6bc85082b3fb7fcfa697a8a
anym@anym:~/SDN_DeLab$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS          NAMES
1115a45eaaae   delab    "python script.py"      15 seconds ago Up 9 seconds          sdn
anym@anym:~/SDN_DeLab$ sudo docker exec -it sdn /bin/sh
/DeLab # ls
Dockerfile      Inventario.csv  README.md        requirements.txt  script.py
/DeLab # cat Inventario.csv
Tipo_Producto,Modelo,Nombre,Direccion_MAC,Serial,Direccion_IP_LAN,Direccion_IP_Publica,Status
wireless,MR84,Alex's MR84 - 1,e0:55:3d:10:56:8a,Q2EK-2LYB-PCZP,,75.187.61.126,dormant
wireless,MR84,Vegas Living Room MR84,e0:55:3d:10:5a:ca,Q2EK-3UBE-RRUY,192.168.0.20,71.222.80.198,dormant
wireless,MR84,,e0:55:3d:10:5b:d8,Q2EK-ACGE-URXL,,,dormant
wireless,MR84,Vegas Balcony MR84,e0:55:3d:10:5c:48,Q2EK-D4XP-235S,,71.222.80.198,dormant
wireless,MR84,Sun Room,e0:55:3d:10:5e:b2,Q2EK-UKGM-XSD9,192.168.1.95,76.250.206.183,online
wireless,MR32,Alex's MR32,e0:55:3d:b8:3e:70,Q2JD-7RNY-EB7Z,192.168.1.16,71.79.148.43,dormant
wireless,MR42,ap01-dl2,e0:cb:bc:8c:1f:fe,Q2KD-KWMU-7U92,192.168.128.3,76.250.206.183,online
wireless,MR52,Basement AP,e0:55:3d:c0:73:56,Q2LD-3Y7V-7Y2X,192.168.1.94,76.250.206.183,online
wireless,MR52,ap01-dl3,0c:8d:db:b2:2f:2c,Q2LD-D932-NRPU,192.168.1.142,76.250.206.183,online
wireless,MR52,ap01-dl1,0c:8d:db:b2:77:f8,Q2LD-FGN3-VP7S,192.168.128.7,76.250.206.183,online
wireless,MR52,1st Floor AP,e0:55:3d:c0:76:f4,Q2LD-GYL3-KEHX,192.168.1.91,76.250.206.183,online
wireless,MR52,,0c:8d:db:b2:8a:5a,Q2LD-N2U5-D83H,24.144.215.84,24.144.215.84,dormant
wireless,MR52,,0c:8d:db:b2:8c:f0,Q2LD-X2S2-AG2U,,,dormant
wireless,MR52,Office AP,e0:55:3d:c0:72:d8,Q2LD-ZWCZ-UA77,192.168.1.154,76.250.206.183,online
appliance,MX65,MX65,e0:55:3d:73:05:4d,Q2QN-FD4H-JKYA,,76.250.206.183,online
appliance,MX65,mx01-dl1,0c:8d:db:b0:c2:dc,Q2QN-Q6EY-NP7J,,76.250.206.183,online
appliance,MX65,mx01-dl2,0c:8d:db:b0:c3:44,Q2QN-UTMQ-ZJQA,,76.250.206.183,online
appliance,MX65,,0c:8d:db:b0:c4:55,Q2QN-Y5ED-PS7W,,,dormant

```

Figura 7: Comando run interactivo en segundo plano para ejecutar contenedor de imagen delab.

Para conservar los datos creados con el script se debe crear un volumen, que es una carpeta compartida entre el contenedor y el host. Se pueden crear volúmenes de varias

maneras, con el comando `docker volumen create`, a partir del Dockerfile o crear un volumen en el directorio host deseado. Cada una de estas maneras tiene ventajas y desventajas, al utilizar el comando **volumen create** es rápido y sencillo, pero la dirección es un poco difícil de encontrar, al crearlo en el directorio del host no se le puede dar nombre al volumen ni automatizarlo con el Dockerfile y al crear el volumen con un archivo Docker se automatiza el proceso, pero no se le puede asignar nombre ni carpeta host. Por lo tanto, en este caso se utilizó la opción más sencilla, a partir del comando **volumen create** y se verifica la ruta con el comando **docker volumen inspect**

```
anym@anym:~/SDN_DeLab$ sudo docker volume create sdn-vol
sdn-vol
anym@anym:~/SDN_DeLab$ sudo docker volume ls
DRIVER      VOLUME NAME
local       sdn-vol
anym@anym:~/SDN_DeLab$ sudo docker inspect sdn-vol
[
  {
    "CreatedAt": "2022-07-31T13:24:11-04:00",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "/var/lib/docker/volumes/sdn-vol/_data",
    "Name": "sdn-vol",
    "Options": {},
    "Scope": "local"
  }
]
```

Figura 8: Comando para crear volumen sdn-vol

Para utilizar el volumen en un contenedor, se utiliza el comando **docker run -it -d -name [container name] -v [volumen name]:[container directory] images**. Se verifica que está corriendo el contenedor y con el comand **docker inspect [container name]** en el segmento Mounts se verifica la dirección del volumen.

```
anym@anym:~/SDN_DeLab$ sudo docker run -it -d --name sdn -v sdn-vol:/DeLab delab
307581fa11260950cdf7590a0b6bb48f6e4b57fcb30f524bc2c979023daccffe
anym@anym:~/SDN_DeLab$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS          NAMES
307581fa1126   delab    "python script.py"      6 seconds ago Up 4 seconds   sdn
anym@anym:~/SDN_DeLab$ sudo docker inspect sdn
[
  {
    "Id": "307581fa11260950cdf7590a0b6bb48f6e4b57fcb30f524bc2c979023daccffe",
```

Figura 9: Comando para crear volumen sdn-vol

```

},
"Mounts": [
  {
    "Type": "volume",
    "Name": "sdn-vol",
    "Source": "/var/lib/docker/volumes/sdn-vol/_data",
    "Destination": "/DeLab",
    "Driver": "local",
    "Mode": "z",
    "RW": true,
    "Propagation": ""
  }
]

```

Figura 10: Ruta del volumen sdn-vol

Se accede a la ruta del volumen y se observa que está el archivo Inventario.csv, como se muestra en la Figura 11.

```

amyn@amyn:~$ sudo ls /var/lib/docker/volumes/sdn-vol/_data
[sudo] contraseña para amyn:
Dockerfile Inventario.csv README.md requirements.txt script.py
amyn@amyn:~$ sudo cat /var/lib/docker/volumes/sdn-vol/_data/Inventario.csv
Tipo_Producto,Modelo,Nombre,Direccion_MAC,Serial,Direccion_IP_LAN,Direccion_IP_Publica,Status
wireless,MR84,Alex's MR84 - 1,e0:55:3d:10:56:8a,Q2EK-2LYB-PCZP,,75.187.61.126,dormant
wireless,MR84,Vegas Living Room MR84,e0:55:3d:10:5a:ca,Q2EK-3UBE-RRUY,192.168.0.20,71.222.80.198,dormant
wireless,MR84,,e0:55:3d:10:5b:d8,Q2EK-ACGE-URXL,,,dormant
wireless,MR84,Vegas Balcony MR84,e0:55:3d:10:5c:48,Q2EK-D4XP-235S,,71.222.80.198,dormant
wireless,MR84,Sun Room,e0:55:3d:10:5e:b2,Q2EK-UKGM-XSD9,192.168.1.95,76.250.206.183,online
wireless,MR32,Alex's MR32,e0:55:3d:b8:3e:70,Q2JD-7RNY-EB7Z,192.168.1.16,71.79.148.43,dormant
wireless,MR42,ap01-dl2,e0:cb:bc:8c:1f:fe,Q2KD-KWMU-7U92,192.168.128.3,76.250.206.183,online
wireless,MR52,Basement AP,e0:55:3d:c0:73:56,Q2LD-3Y7V-7Y2X,192.168.1.94,76.250.206.183,online
wireless,MR52,ap01-dl3,0c:8d:db:b2:2f:2c,Q2LD-D932-NRPU,192.168.1.142,76.250.206.183,online
wireless,MR52,ap01-dl1,0c:8d:db:b2:77:f8,Q2LD-FGN3-VP7S,192.168.128.7,76.250.206.183,online
wireless,MR52,1st Floor AP,e0:55:3d:c0:76:f4,Q2LD-GYL3-KEHX,192.168.1.91,76.250.206.183,online
wireless,MR52,,0c:8d:db:b2:8a:5a,Q2LD-N2U5-D83H,24.144.215.84,24.144.215.84,dormant
wireless,MR52,,0c:8d:db:b2:8c:f0,Q2LD-X2S2-AG2U,,,dormant
wireless,MR52,Office AP,e0:55:3d:c0:72:d8,Q2LD-ZWCZ-UA77,192.168.1.154,76.250.206.183,online
appliance,MX65,MX65,e0:55:3d:73:05:4d,Q2QN-FD4H-JKYA,,76.250.206.183,online
appliance,MX65,mx01-dl1,0c:8d:db:b0:c2:dc,Q2QN-Q6EY-NP7J,,76.250.206.183,online
appliance,MX65,mx01-dl2,0c:8d:db:b0:c3:44,Q2QN-UTMQ-ZJQA,,76.250.206.183,online
appliance,MX65,,0c:8d:db:b0:c4:55,Q2QN-Y5ED-P57W,,,dormant

```

Figura 11: Verificación del archivo Inventario.csv en la ruta del volumen sdn-vol

Ahora, se crea un archivo docker-compose.yml, este permite configurar los servicios de las aplicaciones. En este se especifica el nombre del contenedor, el nombre de la imagen, el volumen y el directorio del contenedor. Y con la opción external: true se referencia un volumen creado externamente. El docker-compose.yml creado se muestra en la Figura 12.

```

SDN_DeLab > docker-compose.yml
1  services:
2    sdn:
3      image: delab
4      volumes:
5        - sdn-vol:/DeLab
6  volumes:
7    sdn-vol:
8      external: true

```

Figura 12: Archivo docker-compose.yml

Se utiliza el comando **docker compose up -d** para correr el contenedor en segundo plano, y se verifica que está corriendo.

```

anym@anym:~/SDN_DeLab$ sudo docker compose up -d
[+] Running 1/1
  Container sdn_delab-sdn-1 Started
anym@anym:~/SDN_DeLab$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS          NAMES
343985370ee9   delab     "python script.py"      35 seconds ago Up 30 seconds          sdn_delab-sdn-1

```

Figura 13: Comando docker compose up -d

```

anym@anym:~/SDN_DeLab$ sudo docker exec -it sdn_delab-sdn-1 /bin/sh
/DeLab # ls
Dockerfile      Inventario.csv  README.md       requirements.txt  script.py
/DeLab # cat Inventario.csv
Tipo_Producto,Modelo,Nombre,Direccion_MAC,Serial,Direccion_IP_LAN,Direccion_IP_Publica,Status
wireless,MR84,Alex's MR84 - 1,e0:55:3d:10:56:8a,Q2EK-2LYB-PCZP,,75.187.61.126,dormant
wireless,MR84,Vegas Living Room MR84,e0:55:3d:10:5a:ca,Q2EK-3UBE-RRUV,192.168.0.20,71.222.80.198,dormant
wireless,MR84,,e0:55:3d:10:5b:d8,Q2EK-ACGE-URXL,,,dormant
wireless,MR84,Vegas Balcony MR84,e0:55:3d:10:5c:48,Q2EK-D4XP-235S,,71.222.80.198,dormant
wireless,MR84,Sun Room,e0:55:3d:10:5e:b2,Q2EK-UKGM-XSD9,192.168.1.95,76.250.206.183,online
wireless,MR32,Alex's MR32,e0:55:3d:b8:3e:70,Q2JD-7RNY-EB7Z,192.168.1.16,71.79.148.43,dormant
wireless,MR42,ap01-dl2,e0:cb:bc:8c:1f:fe,Q2KD-KWMU-7U92,192.168.128.3,76.250.206.183,online
wireless,MR52,Basement AP,e0:55:3d:c0:73:56,Q2LD-3Y7V-7Y2X,192.168.1.94,76.250.206.183,online
wireless,MR52,ap01-dl3,0c:8d:db:b2:2f:2c,Q2LD-D932-NRPU,192.168.1.142,76.250.206.183,online
wireless,MR52,ap01-dl1,0c:8d:db:b2:77:f8,Q2LD-FGN3-VP7S,192.168.128.7,76.250.206.183,online
wireless,MR52,1st Floor AP,e0:55:3d:c0:76:f4,Q2LD-GYL3-KEHX,192.168.1.91,76.250.206.183,online
wireless,MR52,,0c:8d:db:b2:8a:5a,Q2LD-N2U5-D83H,24.144.215.84,24.144.215.84,dormant
wireless,MR52,,0c:8d:db:b2:8c:f0,Q2LD-X2S2-AG2U,,,dormant
wireless,MR52,Office AP,e0:55:3d:c0:72:d8,Q2LD-ZWCZ-UA77,192.168.1.154,76.250.206.183,online
appliance,MX65,MX65,e0:55:3d:73:05:4d,Q2QN-FD4H-JKYA,,76.250.206.183,online
appliance,MX65,MX01-dl1,0c:8d:db:b0:c2:dc,Q2QN-Q6EY-NP7J,,76.250.206.183,online
appliance,MX65,MX01-dl2,0c:8d:db:b0:c3:44,Q2QN-UTMQ-ZJQA,,76.250.206.183,online
appliance,MX65,,0c:8d:db:b0:c4:55,Q2QN-YSED-P57W,,,dormant
/DeLab #

```

Figura 14: Verificación de archivo Inventario.csv en la ruta del contenedor /DeLab

Luego, se verifica la ruta del volumen asociado es este contenedor con el comando **docker inspect sdn_delab-sdn-1**


```
anym@anym:~/SDN_DeLab$ sudo docker inspect sdn_delab-sdn-1
[
  {
    "Id": "343985370ee97cb1137f52dbd04d5a89735ff0e376714aa2dc2f324759688377",
    "Created": "2022-07-31T17:40:12.185643484Z",
    "Path": "python",
    "Args": [
      "script.py"
    ],
  },
]
```

Figura 15: Ubicación del volumen utilizado para contenedor sdn_delab-sdn-1

```
    "Mounts": [
      {
        "Type": "volume",
        "Name": "sdn-vol",
        "Source": "/var/lib/docker/volumes/sdn-vol/_data",
        "Destination": "/DeLab",
        "Driver": "local",
        "Mode": "z",
        "RW": true,
        "Propagation": ""
      }
    ]
  }
}
```

Figura 16: Verificación de la ruta del volumen

Finalmente, se accede a la ruta del volumen y se accede al archivo Inventario.csv.

```
anym@anym:~$ sudo ls /var/lib/docker/volumes/sdn-vol/_data
Dockerfile Inventario.csv README.md requirements.txt script.py
anym@anym:~$ sudo cat /var/lib/docker/volumes/sdn-vol/_data/Inventario.csv
Tipo_Producto,Modelo,Nombre,Direccion_MAC,Serial,Direccion_IP_LAN,Direccion_IP_Publica,Status
wireless,MR84,Alex's MR84 - 1,e0:55:3d:10:56:8a,Q2EK-2LYB-PCZP,,75.187.61.126,dormant
wireless,MR84,Vegas Living Room MR84,e0:55:3d:10:5a:ca,Q2EK-3UBE-RRUY,192.168.0.20,71.222.80.198,dormant
wireless,MR84,,e0:55:3d:10:5b:d8,Q2EK-ACGE-URXL,,,dormant
wireless,MR84,Vegas Balcony MR84,e0:55:3d:10:5c:48,Q2EK-D4XP-235S,,71.222.80.198,dormant
wireless,MR84,Sun Room,e0:55:3d:10:5e:b2,Q2EK-UKGM-XSD9,192.168.1.95,76.250.206.183,online
wireless,MR32,Alex's MR32,e0:55:3d:b8:3e:70,Q2JD-7RNY-EB7Z,192.168.1.16,71.79.148.43,dormant
wireless,MR42,ap01-dl2,e0:cb:bc:8c:1f:fe,Q2KD-KWMU-7U92,192.168.128.3,76.250.206.183,online
wireless,MR52,Basement AP,e0:55:3d:c0:73:56,Q2LD-3Y7V-7Y2X,192.168.1.94,76.250.206.183,online
wireless,MR52,ap01-dl3,0c:8d:db:b2:2f:2c,Q2LD-D932-NRPU,192.168.1.142,76.250.206.183,online
wireless,MR52,ap01-dl1,0c:8d:db:b2:77:f8,Q2LD-FGN3-VP7S,192.168.128.7,76.250.206.183,online
wireless,MR52,1st Floor AP,e0:55:3d:c0:76:f4,Q2LD-GYL3-KEHX,192.168.1.91,76.250.206.183,online
wireless,MR52,,0c:8d:db:b2:8a:5a,Q2LD-N2US-D83H,24.144.215.84,24.144.215.84,dormant
wireless,MR52,,0c:8d:db:b2:8c:f0,Q2LD-X2S2-AG2U,,,dormant
wireless,MR52,Office AP,e0:55:3d:c0:72:d8,Q2LD-ZWCZ-UA77,192.168.1.154,76.250.206.183,online
appliance,MX65,MX65,e0:55:3d:73:05:4d,Q2QN-FD4H-JKYA,,76.250.206.183,online
appliance,MX65,mx01-dl1,0c:8d:db:b0:c2:dc,Q2QN-Q6EY-NP7J,,76.250.206.183,online
appliance,MX65,mx01-dl2,0c:8d:db:b0:c3:44,Q2QN-UTMQ-ZJQA,,76.250.206.183,online
appliance,MX65,,0c:8d:db:b0:c4:55,Q2QN-Y5ED-P57W,,,dormant
```

Figura 17: Verificación de archivo Inventario.csv en la ruta del volumen sdn-vol

Como último paso para esta práctica se realiza un commit para agregar el archivo README.md modificado, el archivo docker-compose.yml y el archivo Practica III.pdf con los comandos ejecutados para esta práctica. Como se indica en la siguiente Figura.


```
anyn@anyn:~/SDN_DeLab$ git status
En la rama master
Tu rama está actualizada con 'origin/master'.

Cambios no rastreados para el commit:
  (usa "git add <archivo>..." para actualizar lo que será confirmado)
  (usa "git checkout -- <archivo>..." para descartar los cambios en el directorio de trabajo)

      modificado:   README.md

Archivos sin seguimiento:
  (usa "git add <archivo>..." para incluirlo a lo que se será confirmado)

      Practica III.pdf
      docker-compose.yml

sin cambios agregados al commit (usa "git add" y/o "git commit -a")
anyn@anyn:~/SDN_DeLab$ git commit -a
[master cb00a14] Se añade archivo docker-compose.yml que permite configurar los servicios de las aplicaciones para ejecutar un contenedor.
1 file changed, 17 insertions(+)
```

Figura 18: Commit de los últimos archivos de práctica III