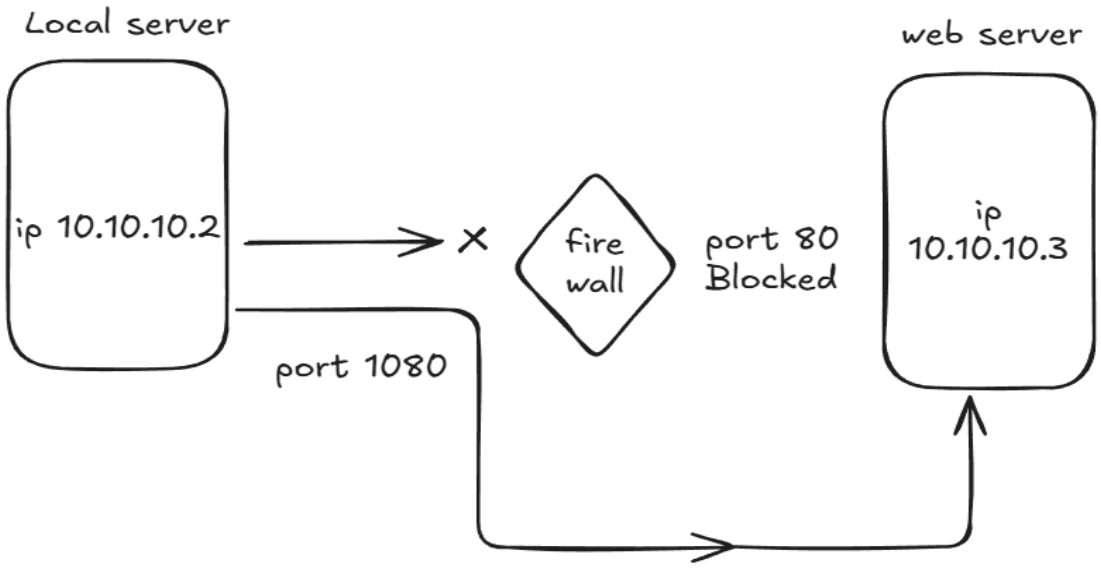
Dynamic Port Forwarding with SSH

Purpose:
Create a SOCKS proxy using SSH to allow applications (Browser, Curl) to route traffic dynamically to any destination through an encrypted SSH tunnel.

Lab



SSH tunnel

Components

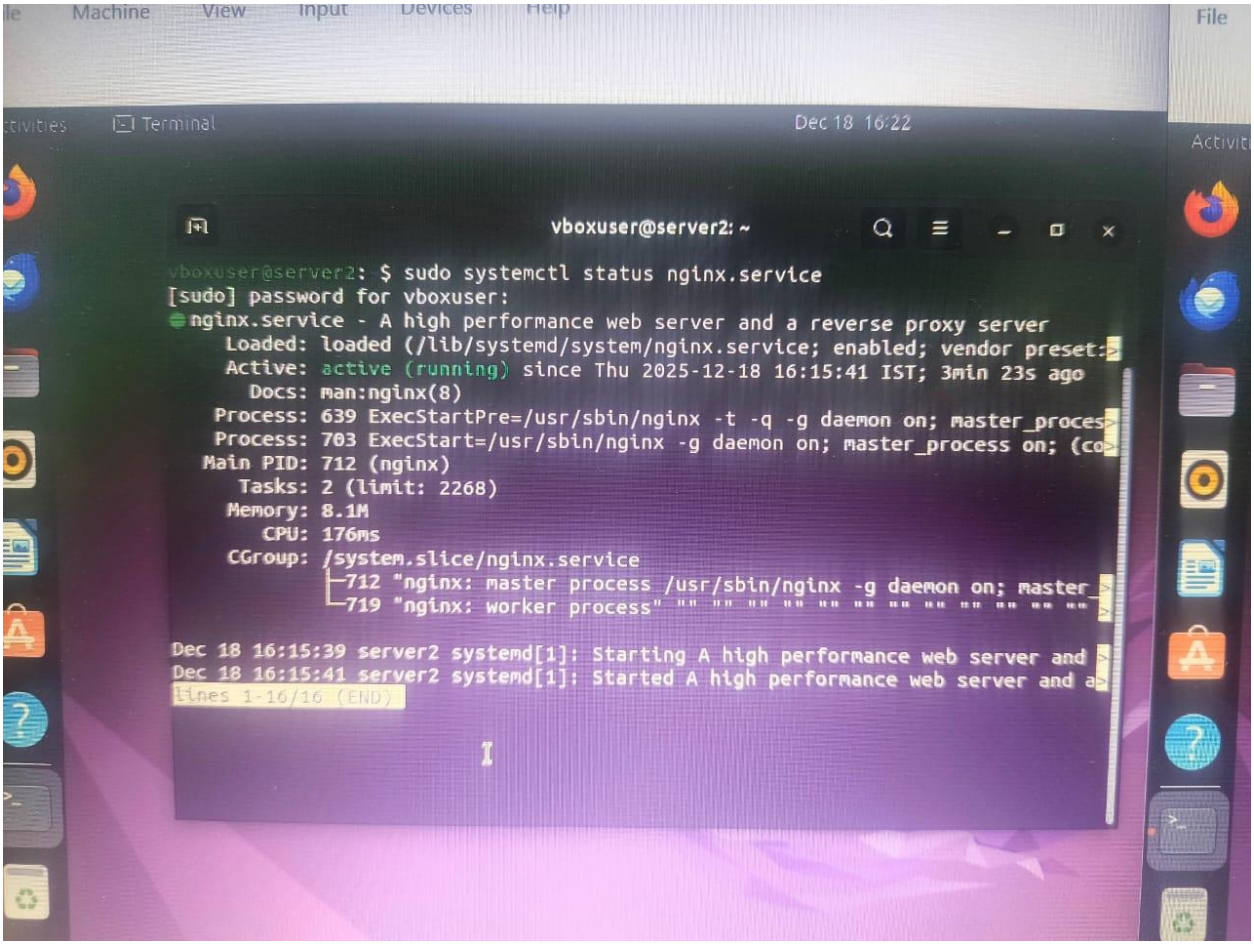
Component	Description
VM1	Local server (Client)
VM2	Remote Web server
Nginx	Web service on port 80
SSH	Service on port 22
Firewall	Port 80 blocked on Web server

Prerequisites

- SSH access to remote server
- Web service running on remote server
- SSH service running
- Port 80 blocked on Web server

Steps

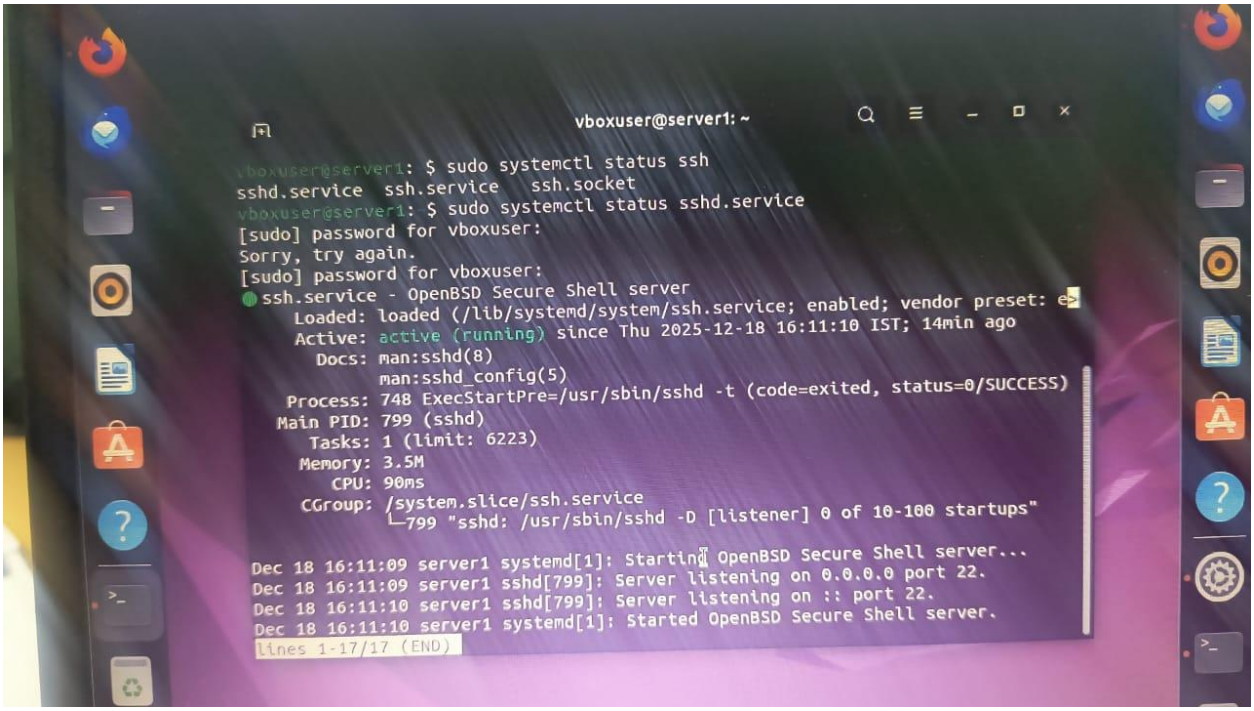
Step 1: Verify Web Service on Client VM (10.10.1.3)




```
sudo systemctl status nginx
# OR
sudo systemctl status apache2
```

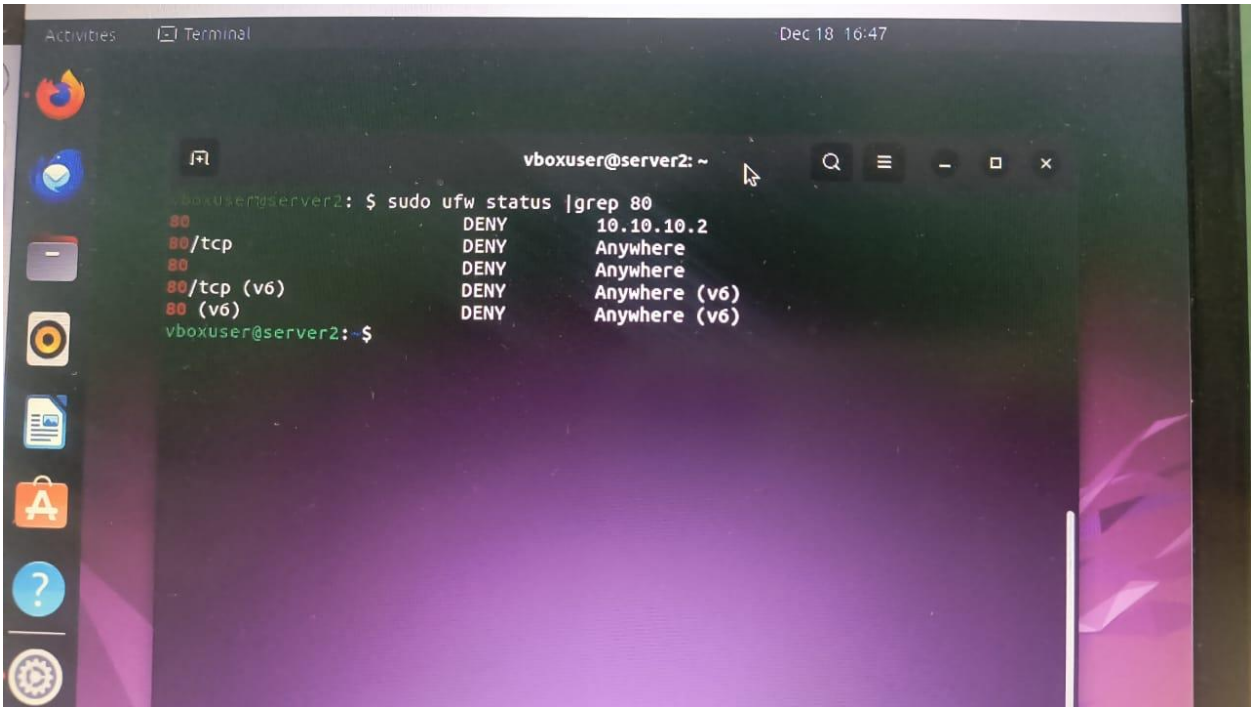
Check that the web service is running.

Step 2: Verify SSH Service on VM (10.10.10.2)



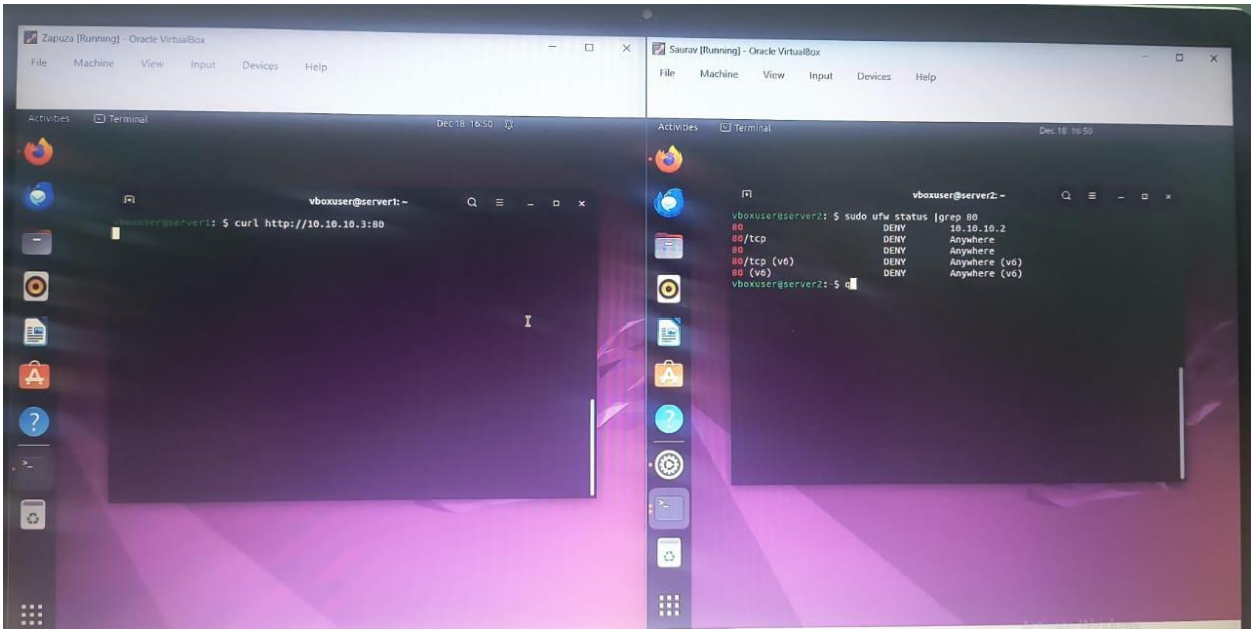

```
sudo systemctl status ssh
# Check listening port
sudo ss -tuln | grep :22
```

Step 3: Verify Port 80 Blocked on Web Server




```
sudo ufw status | grep 80
```

Step 4: Direct Access Test (Expected to Fail)

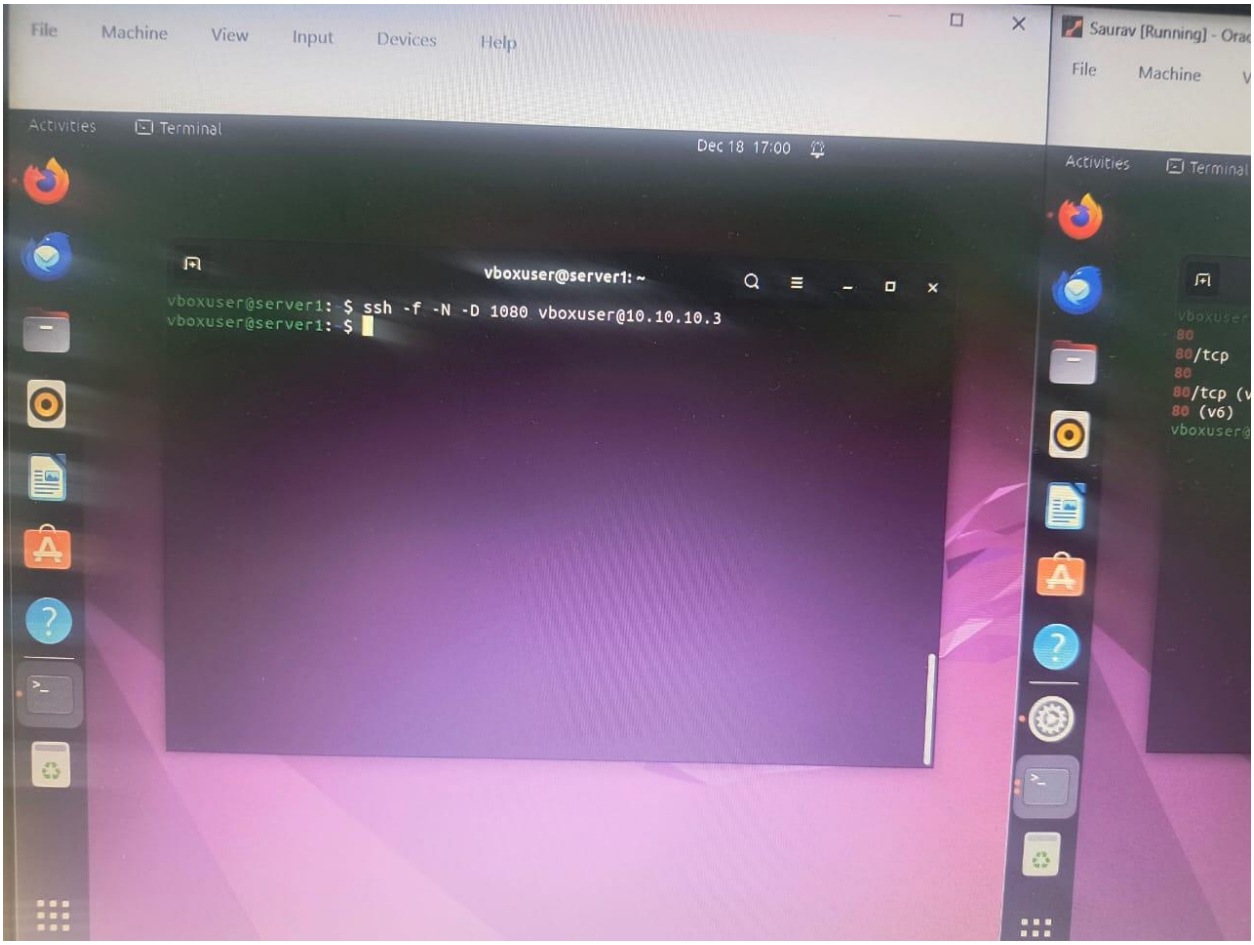


From Server VM (10.10.10.2):

`curl` http://10.10.10.3:80

Fails due to firewall restrictions or localhost binding.

Step 5: Create SSH Dynamic Port Forwarding



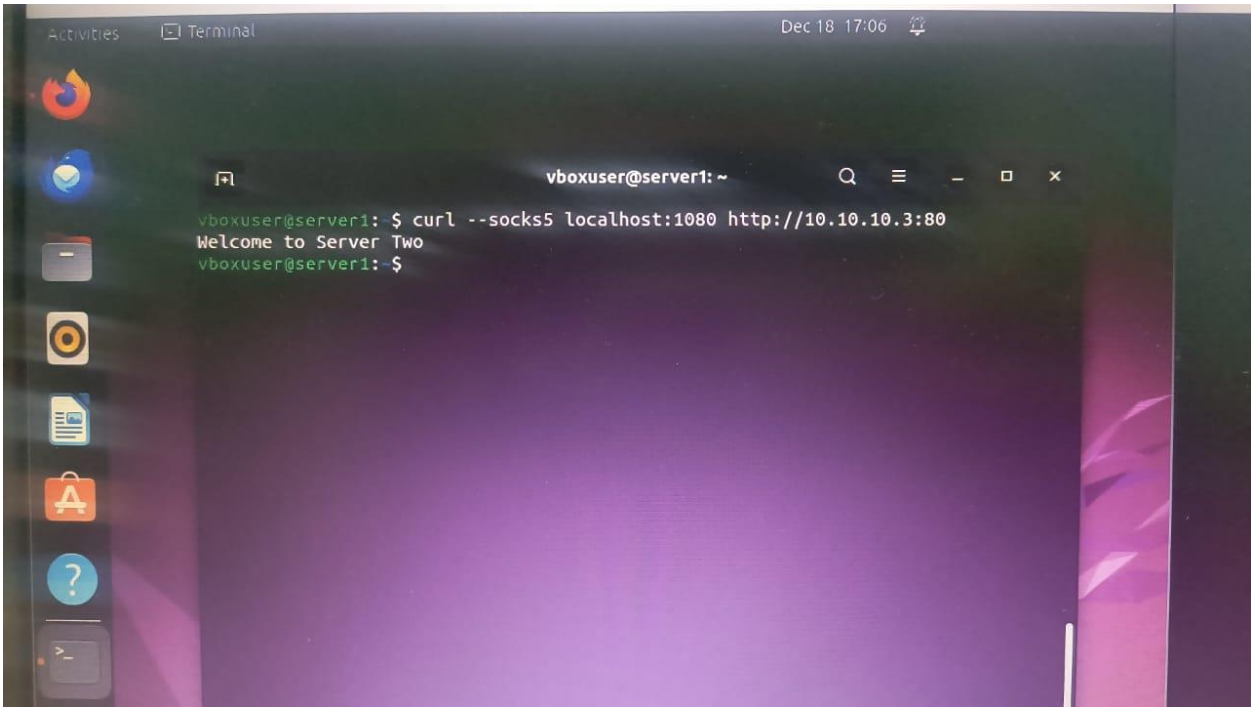
On Client VM (10.10.10.2):

```
ssh -f -N -D 1080 username@Server_IP
```

Flags Explained:

- `-D` → Dynamic port forwarding
- `-f` → Run SSH in background
- `-N` → Do not open remote shell

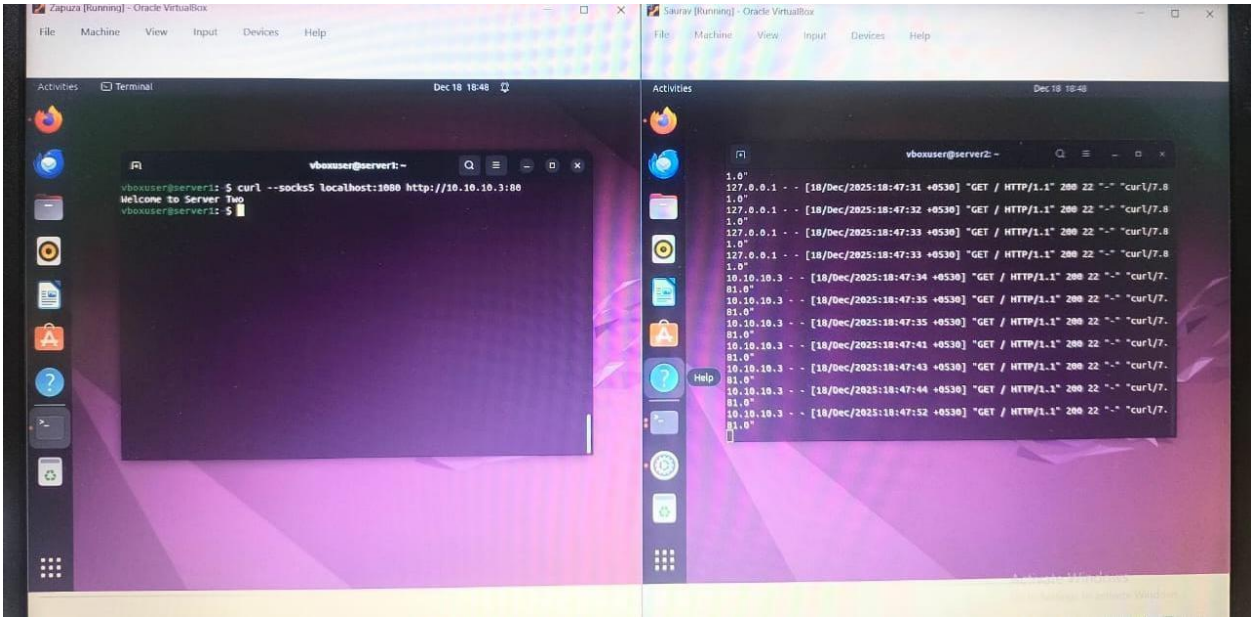
Step 6: Access the Service via Proxy



`curl --socks5 localhost:1080` http://10.10.10.3:80

Or open the browser and set SOCKS5 proxy to `localhost:1080`.

Step 7: Check Web Server Logs on Client VM




```
sudo tail -f /var/log/nginx/access.log
```

Why 127.0.0.1?

Traffic reaches Nginx through the SSH tunnel, which forwards requests locally on the client VM.