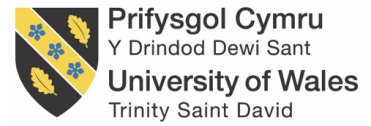


Sarah LEVY
Student - 2422988



NETWORK OPERATING SYSTEMS AND SERVICES

- Configure and manage Virtualised network -

› Coursework 2 ‹

Due on 13-05-25

SUMMARY:

Introduction.....	2
VPN configuration.....	3
1. Windows Server VPN Configuration.....	3
2. Debian Server VPN Configuration.....	6
3. Troubleshooting VPN Issues.....	7
4. Testing VPN connectivity:.....	8
Domain Services Configuration.....	9
1. Swansea Domain.....	9
A. Domain Controller Promotion.....	9
B. DNS Configuration.....	10
C. DHCP Scope Options.....	11
D. Creating a Test User Account.....	12
E. Joining a Client to the Domain.....	13
2. Cardiff Child Domain.....	14
A. Provisioning the Debian Server as a Domain Controller.....	14
B. DNS Integration with Parent Domain.....	15
C. Adjusting Password Policy and Creating a Test User.....	16
D. PAM and Realm Configuration on Ubuntu Client.....	16
E. Verifying Kerberos Authentication and Domain Integration.....	17
F. Establishing Trust Between Domains.....	18
Importing Users.....	19
1. Users in the Swansea Domain.....	19
2. Users in the Cardiff Domain.....	20
Shared Folder Configuration.....	21
1. Adding a New Virtual Disk.....	21
2. Sharing and Configuring permissions.....	23
3. Testing Access Rights.....	23
IIS Web Server Setup.....	25
1. Generating a Self-Signed SSL Certificate.....	25
2. Setting up the Website in IIS.....	25
3. Creating the HTML Page.....	26
4. Verifying the Website on Clients.....	27
Remote Desktop Services (RDS).....	30
1. Installing the RDS Role.....	30
2. Creating the Session Collection and Assigning Users.....	30
3. Publishing Remote Applications.....	31
4. Configuring IIS Redirect.....	32
5. Testing the Access from Clients.....	33
Conclusion.....	34

TASK 1

Introduction

This report is about the implementation of a virtualized network environment as a continuation of the project initiated in Coursework 1. The network was built using a Type 2 hypervisor (VirtualBox) and complies strictly with the specifications detailed in the coursework 2 specifications.

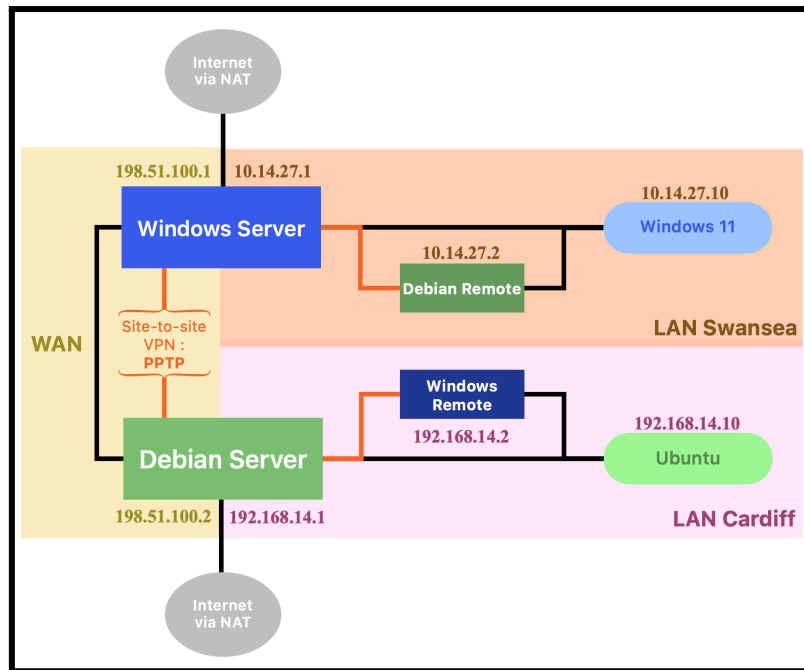
At the end of coursework 1, basic routing, NAT, and DHCP were already functional for both sites:

- **LAN Swansea:** 10.14.27.0/24: contains a Windows Server 2022 (10.14.27.1) and a Windows 11 client (10.14.27.10).
- **LAN Cardiff:** 192.168.14.0/24: contains a Debian Server (192.168.14.1) and an Ubuntu client (192.168.14.10).
- **WAN (bridged):** 198.51.100.0/30: connects the two sites with Windows Server at 198.51.100.1 and Debian Server at 198.51.100.2.

The architecture has been expanded to include:

- A persistent site-to-site VPN
- A Windows Server AD forest (sarah.acca) and a child domain for Cardiff with Samba (car.sarah.acca)
- Domain Trust
- DNS forward and reverse records
- DHCP with advanced options
- A shared folder
- An internal website (webs.sarah.acca)
- And a remote Desktop Services portal (rds.sarah.acca)

Both servers provide internet access to their respective LANs through NAT. A site-to-site VPN was added between the two servers to allow each LAN to access the other through a secure tunnel. The diagram below illustrates this new topology:



[1] Updated network topology

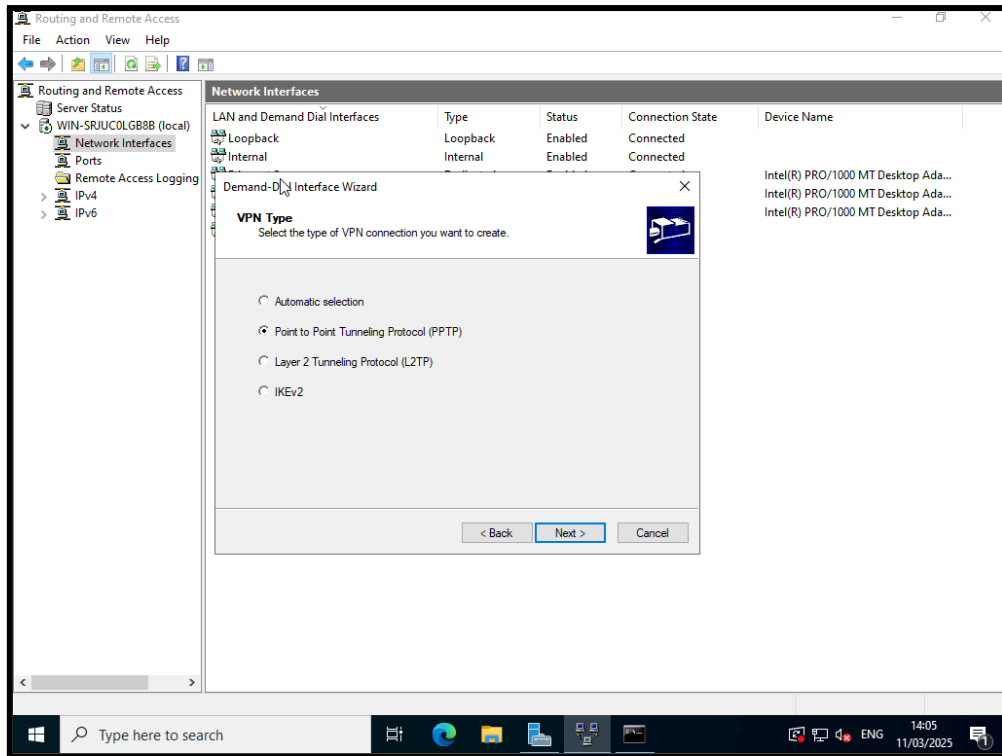
VPN configuration

To ensure inter-site communication between the Swansea and Cardiff networks, a site-to-site VPN was configured using the PPTP protocol.

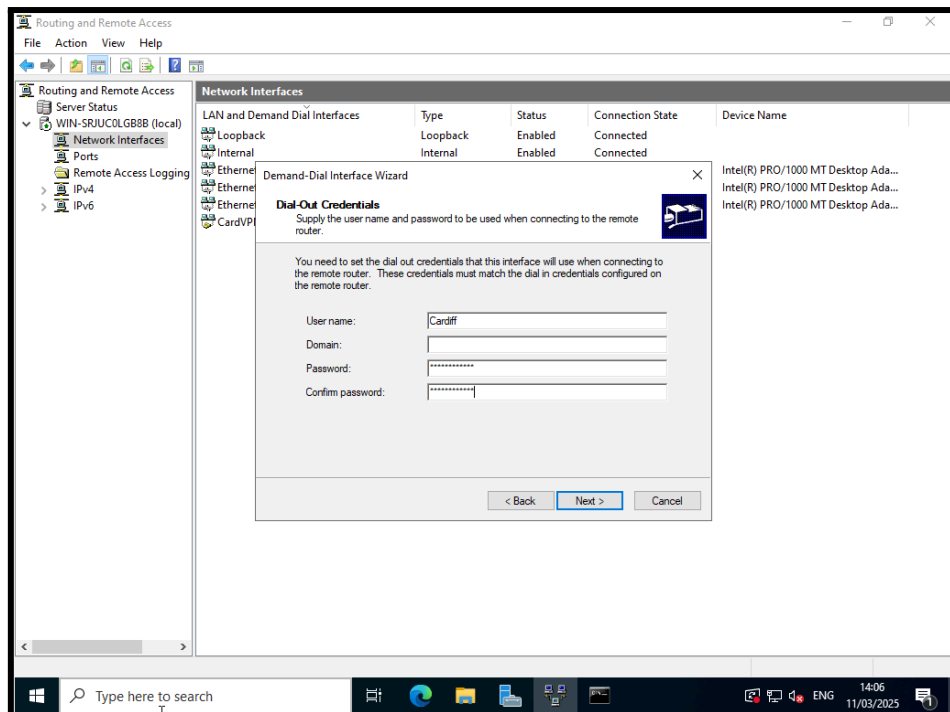
To configure it, it was first necessary to choose which server would initiate the connection. Since the Windows Server hosts the shared folder, the primary Active Directory domain, the web server, and the Remote Desktop Services, it was the most logical choice to take on the role of VPN initiator.

1. Windows Server VPN Configuration

The point-to-point tunneling protocol (PPTP) was chosen for the VPN because it's quick and easy to set up, making it a practical option for this kind of configuration. In the Routing and Remote Access console, a demand-dial interface was created with PPTP as the selected protocol:



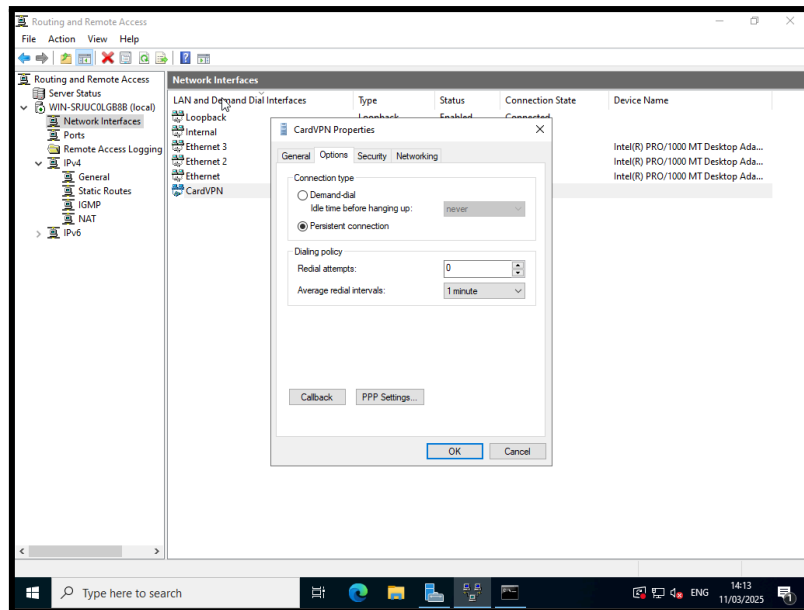
[2] Creation of the VPN interface on Windows Server



[3] Dial-Out Credentials Configuration

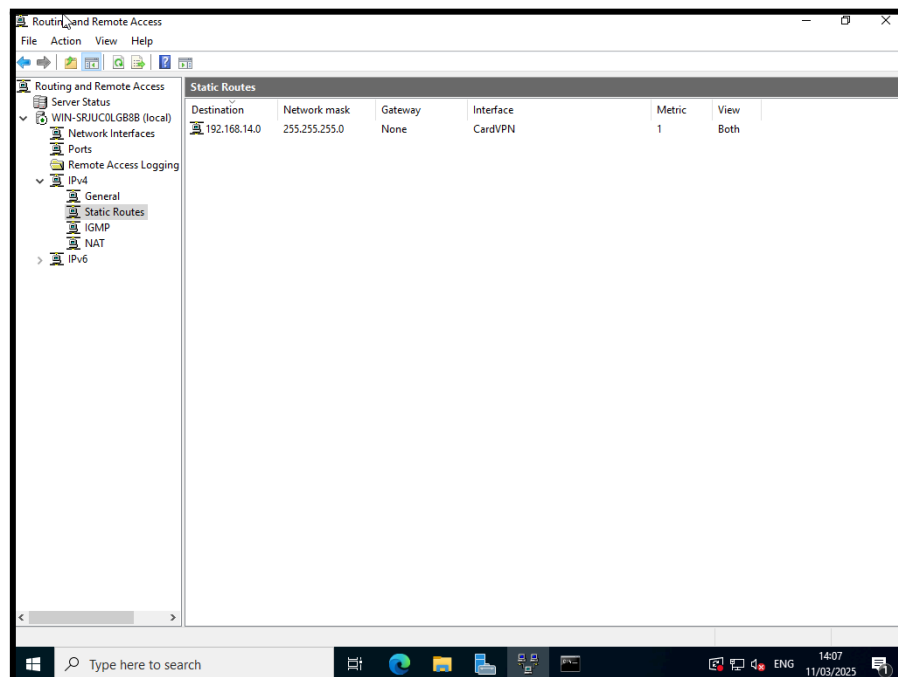
To ensure the connection remains active at all times, the persistent connection option was enabled.

- NETWORK OPERATING SYSTEMS AND SERVICES -

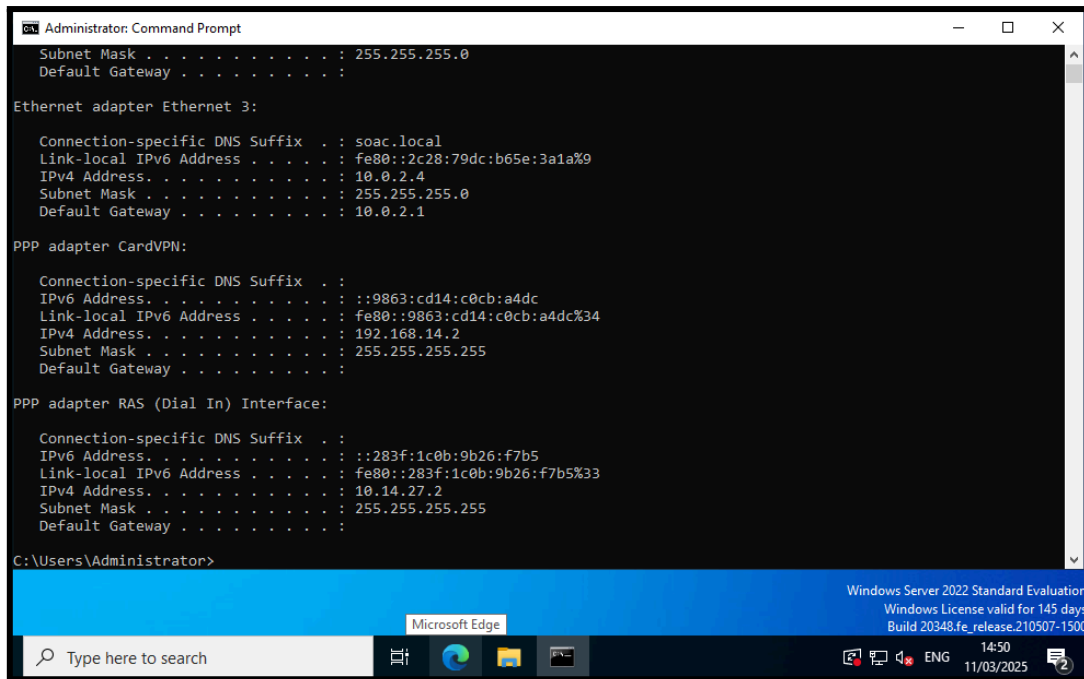


[4] Persistent Connection Settings

Static routes were also added to route all traffic destined for the 192.168.14.0/24 network through the VPN tunnel.



[5] Static route for Cardiff LAN



```
Administrator: Command Prompt
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :

Ethernet adapter Ethernet 3:

Connection-specific DNS Suffix . : soac.local
Link-local IPv6 Address . . . . . : fe80::2c28:79dc:b65e:3a1a%9
IPv4 Address. . . . . : 10.0.2.4
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 10.0.2.1

PPP adapter CardVPN:

Connection-specific DNS Suffix . :
IPv6 Address. . . . . : ::9863:cd14:c0cb:a4dc
Link-local IPv6 Address . . . . . : fe80::9863:cd14:c0cb:a4dc%34
IPv4 Address. . . . . : 192.168.14.2
Subnet Mask . . . . . : 255.255.255.255
Default Gateway . . . . . :

PPP adapter RAS (Dial In) Interface:

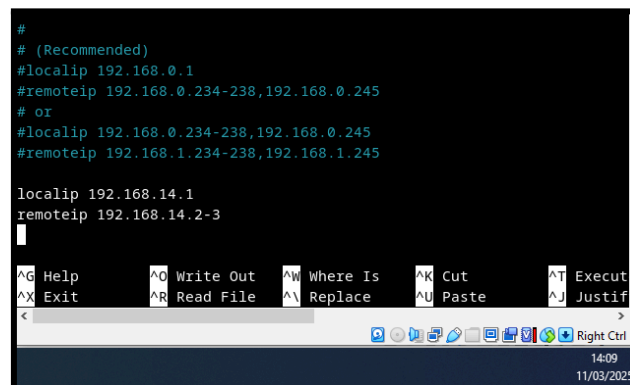
Connection-specific DNS Suffix . :
IPv6 Address. . . . . : ::283f:1c0b:9b26:f7b5
Link-local IPv6 Address . . . . . : fe80::283f:1c0b:9b26:f7b5%33
IPv4 Address. . . . . : 10.14.27.2
Subnet Mask . . . . . : 255.255.255.255
Default Gateway . . . . . :

C:\Users\Administrator>
```

[6] VPN Interface Assigned IP on Windows Server

2. Debian Server VPN Configuration

On the Debian side, the PPTP server was set up to accept connections from the Windows Server. The necessary packages, *ppp* and *pptpd* were installed, and the *pptpd.conf* file was updated to define the local and remote IP addresses.

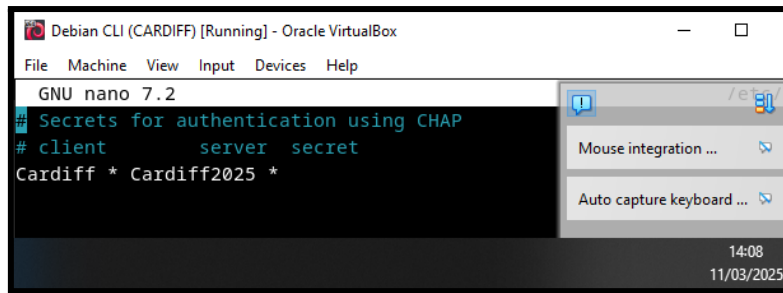


```
#
# (Recommended)
#localip 192.168.0.1
#remoteip 192.168.0.234-238,192.168.0.245
# or
#localip 192.168.0.234-238,192.168.0.245
#remoteip 192.168.1.234-238,192.168.1.245

localip 192.168.14.1
remoteip 192.168.14.2-3
```

[7] pptpd.conf file on Debian

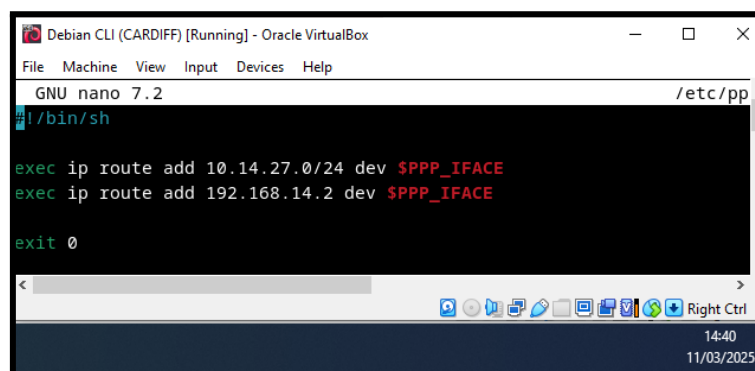
192.168.14.1 is assigned locally and **192.168.14.2-3** is reserved for remote clients so that Windows Server would have the 2nd IP of the scope (.2). Authentication was configured in the chap-secrets file using the credentials matching the ones we chose in the precedent section :



```
Debian CLI (CARDIFF) [Running] - Oracle VirtualBox
File Machine View Input Devices Help
GNU nano 7.2
# Secrets for authentication using CHAP
# client      server  secret
Cardiff * Cardiff2025 *
```

[8] chap-secrets file in debian with the corresponding password.

Additionally, a script was placed in `/etc/ppp/ip-up.d/` to automatically add the appropriate static routes to the remote network whenever the VPN connection is made.



```
Debian CLI (CARDIFF) [Running] - Oracle VirtualBox
File Machine View Input Devices Help
GNU nano 7.2 /etc/pp
! /bin/sh

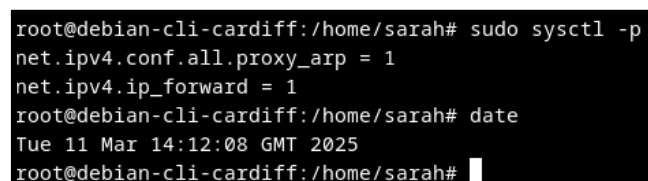
exec ip route add 10.14.27.0/24 dev $PPP_IFACE
exec ip route add 192.168.14.2 dev $PPP_IFACE

exit 0
```

[9] script placed in `/etc/ppp/ip-up.d/` to add static routes

3. Troubleshooting VPN Issues

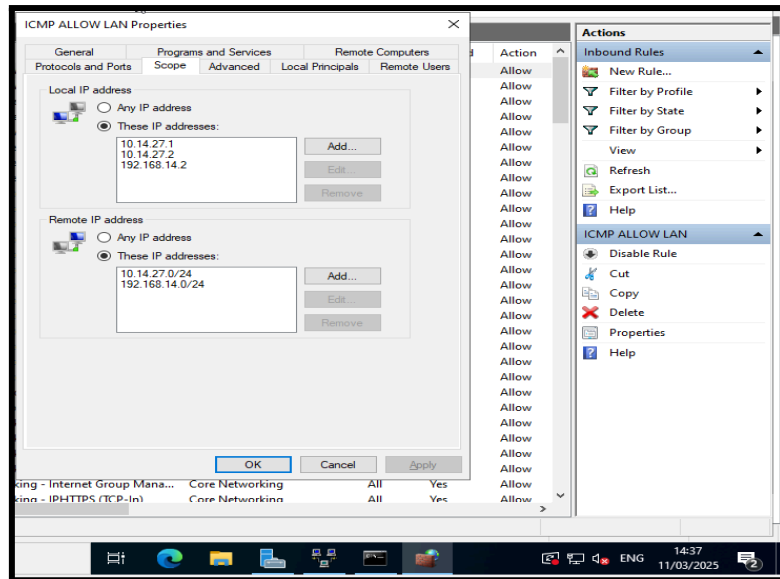
Several issues came up during the implementation. One of them was an IP conflict on the Debian server: both the LAN interface and the VPN interface ended up using the same IP address. To fix this, proxy ARP was enabled for all interfaces using a system configuration command, which allowed the server to properly handle the overlapping addresses.



```
root@debian-cli-cardiff:/home/sarah# sudo sysctl -p
net.ipv4.conf.all.proxy_arp = 1
net.ipv4.ip_forward = 1
root@debian-cli-cardiff:/home/sarah# date
Tue 11 Mar 14:12:08 GMT 2025
root@debian-cli-cardiff:/home/sarah#
```

[10] Enabling proxy ARP and IP Forwarding on Debian

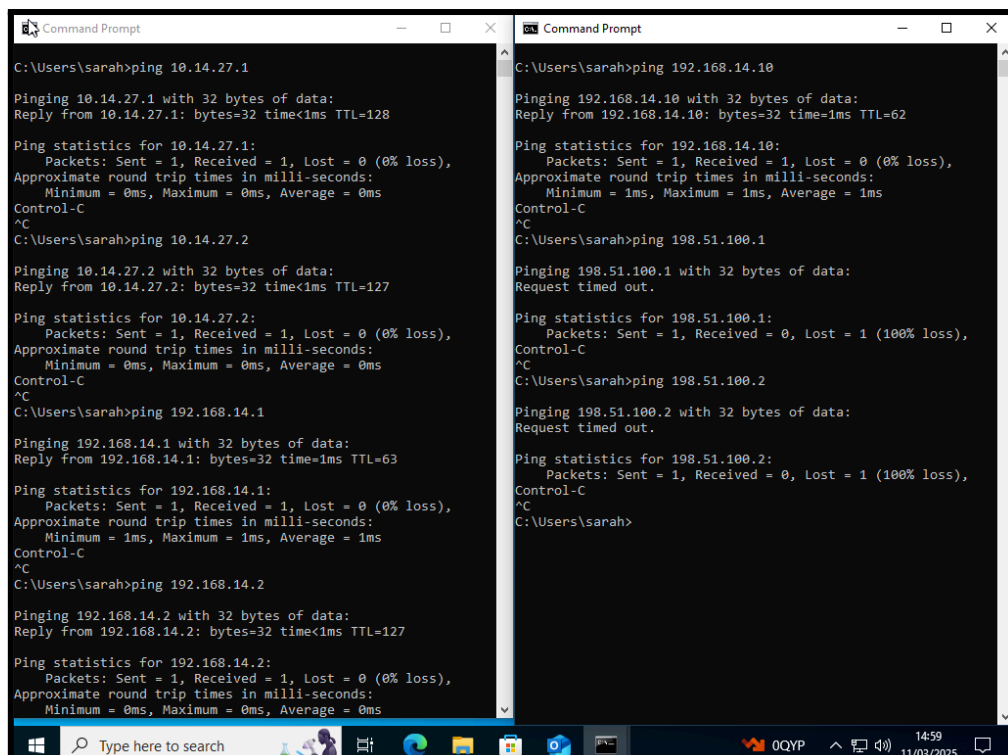
Further modifications were needed to get everything working perfectly. On both servers, firewall rules had to be modified to allow VPN traffic and ICMP requests between the two networks. On the Windows side, specific rules were added to permit ICMP packets coming from the remote subnets `10.14.27.0/24` and `192.168.14.24`.



[11] Firewall Rule modified to allow ICMP from VPN Clients

4. Testing VPN connectivity:

After the configuration was completed and issues were resolved, connectivity was tested by executing ping tests between the devices in the two different LANs.



[12] Successful Ping Tests Between Swansea and Cardiff

The successful ping tests confirmed that the VPN was properly configured and that traffic was correctly routed between the two networks, creating a unified network environment across both sites.

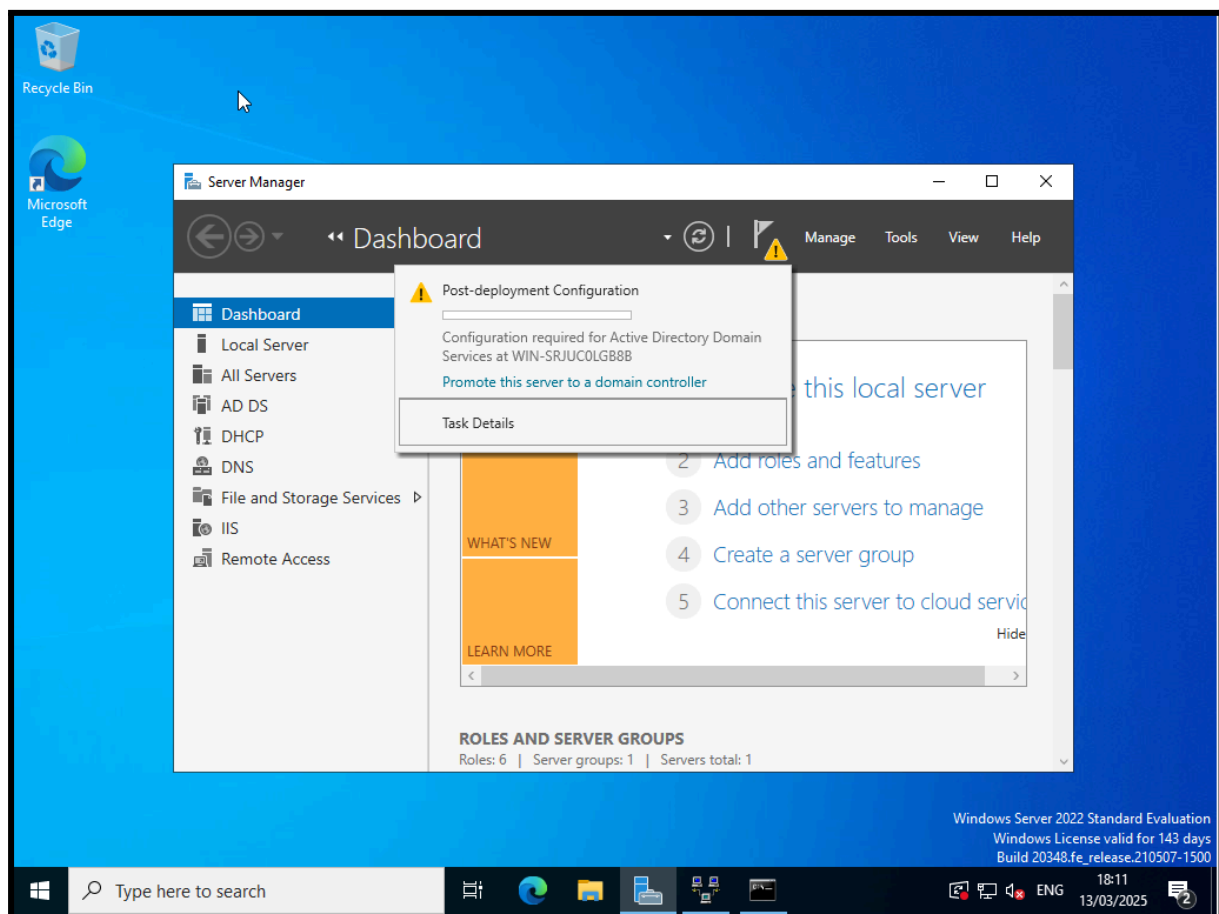
Domain Services Configuration

To implement the domain infrastructure for the Swansea network, the Windows Server machine was configured as a Domain Controller (DC), establishing the root domain **sarah.acca**. This configuration included the setup of Active Directory Domain Services (AD DS), DNS server tuning, DHCP adjustments, and user/domain join validation.

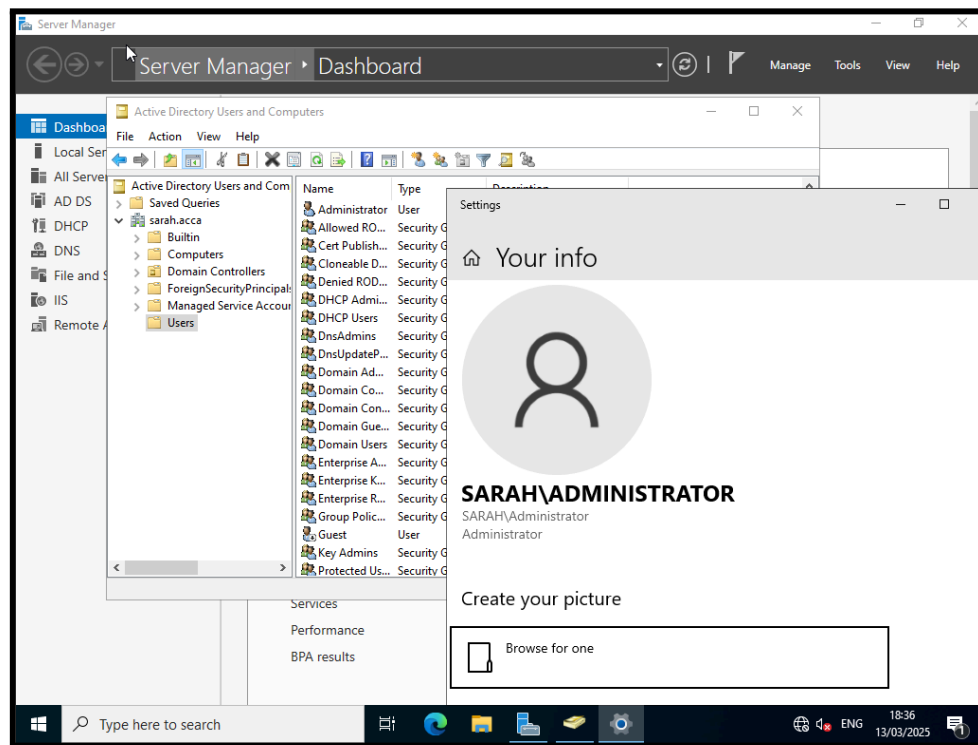
1. Swansea Domain

A. Domain Controller Promotion

After installing the **AD DS** role via Server Manager, the server was promoted to a Domain Controller and a new forest was created under the name **sarah.acca**.



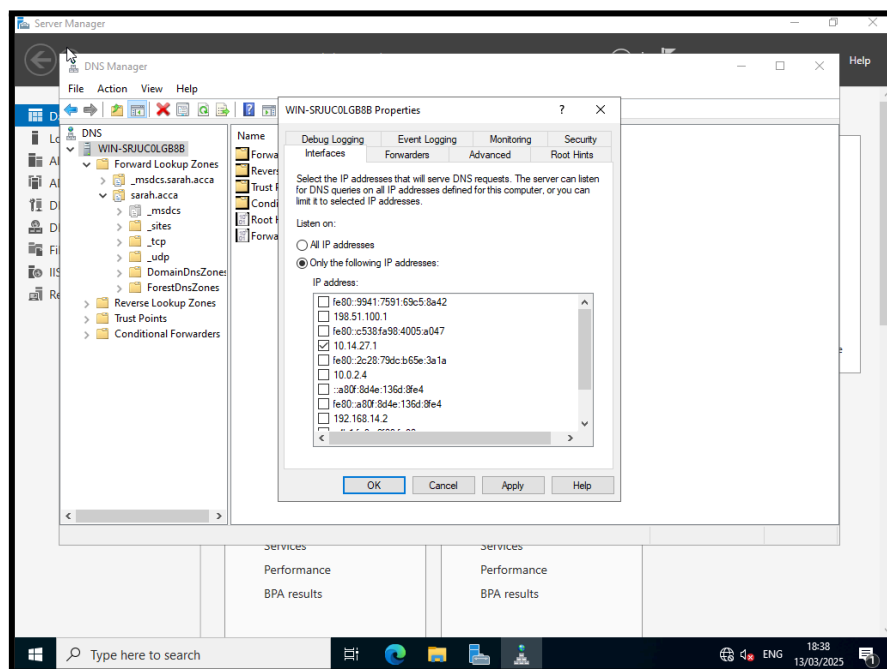
[13] Initial promotion step after AD DS role installation



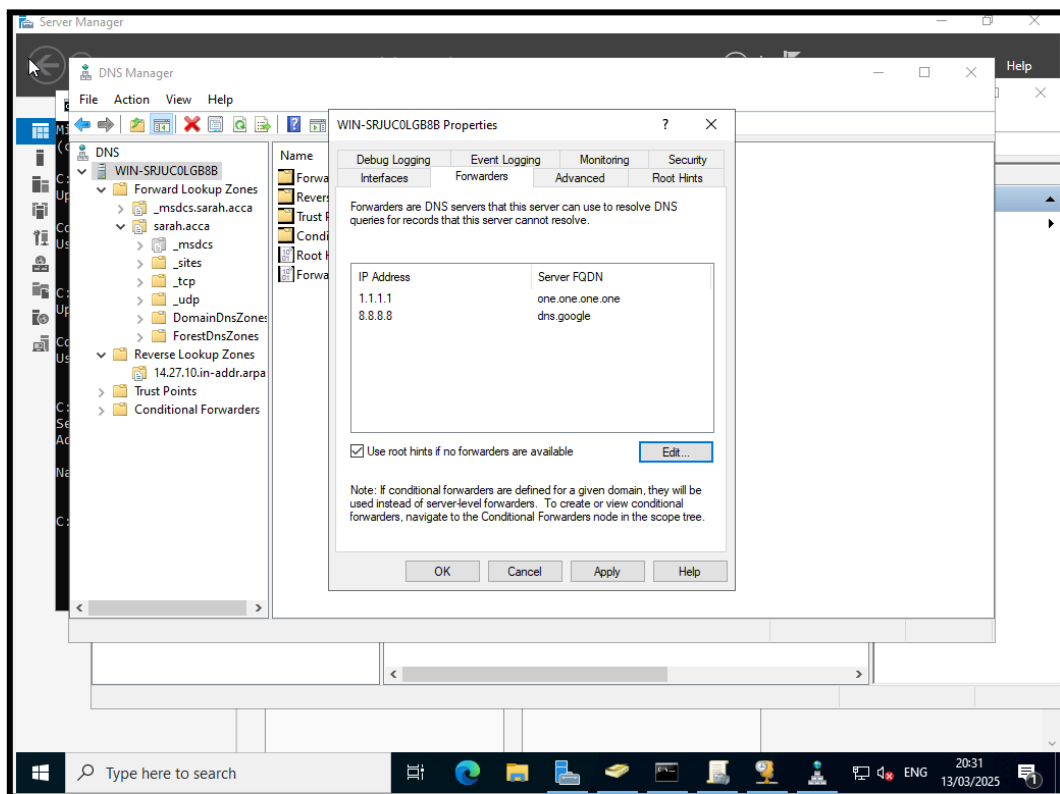
[14] Forest and domain successfully created as sarah.acca.

B. DNS Configuration

Once the domain was created, the DNS service was configured to respond only on appropriate interfaces. Additionally, public DNS forwarders were added to allow external name resolution for domain clients (thus being 1.1.1.1 and 8.8.8.8).



[15] Listening interfaces of the DNS server



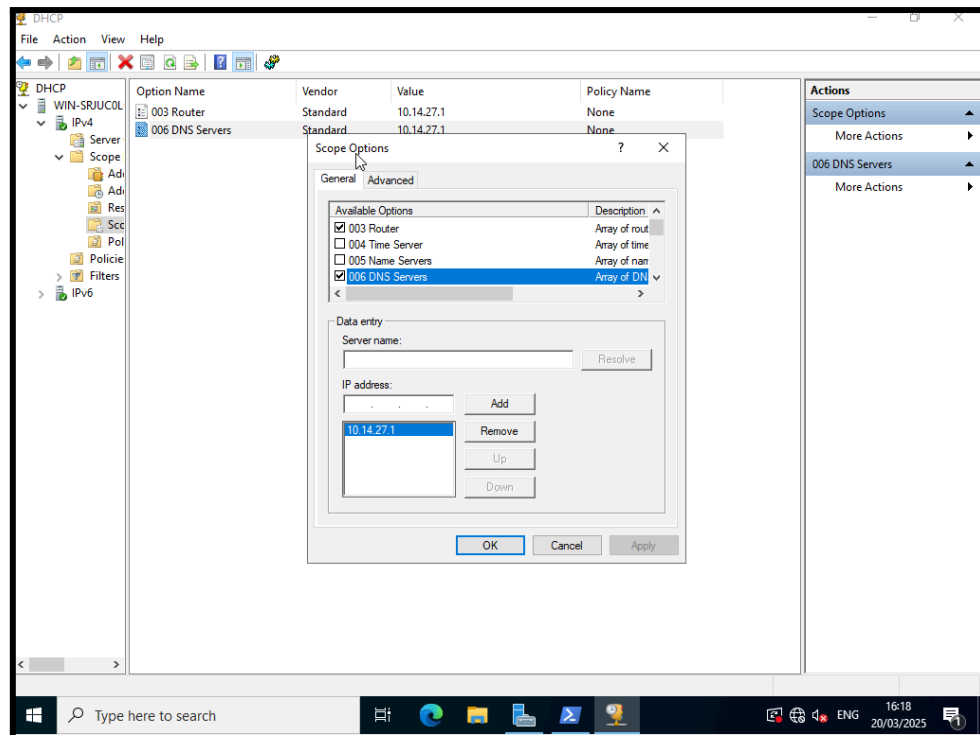
[16] Configuration of external DNS forwarders

The root zone was then edited to remove the unwanted records for unresolved interfaces and to disable dynamic updates (to prevent those records from coming back).

A reverse lookup zone (PTR record) was created for both the Swansea and Cardiff subnets (as visible on the screenshot above too).

C. DHCP Scope Options

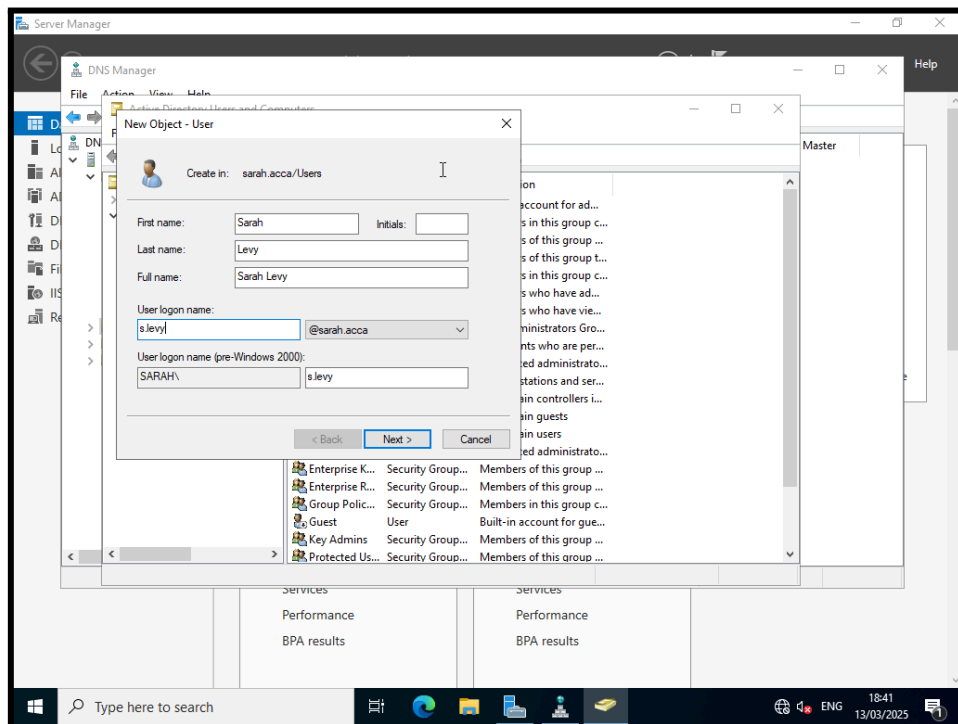
The DHCP server was already installed in Coursework 1. For integration with the new domain, scope options were updated to assign the internal DNS server (10.14.27.1) to clients. This ensures that clients can resolve internal domain names such as *sarah.acca*.



[17] Changing the DNS Server for the DHCP clients

D. Creating a Test User Account

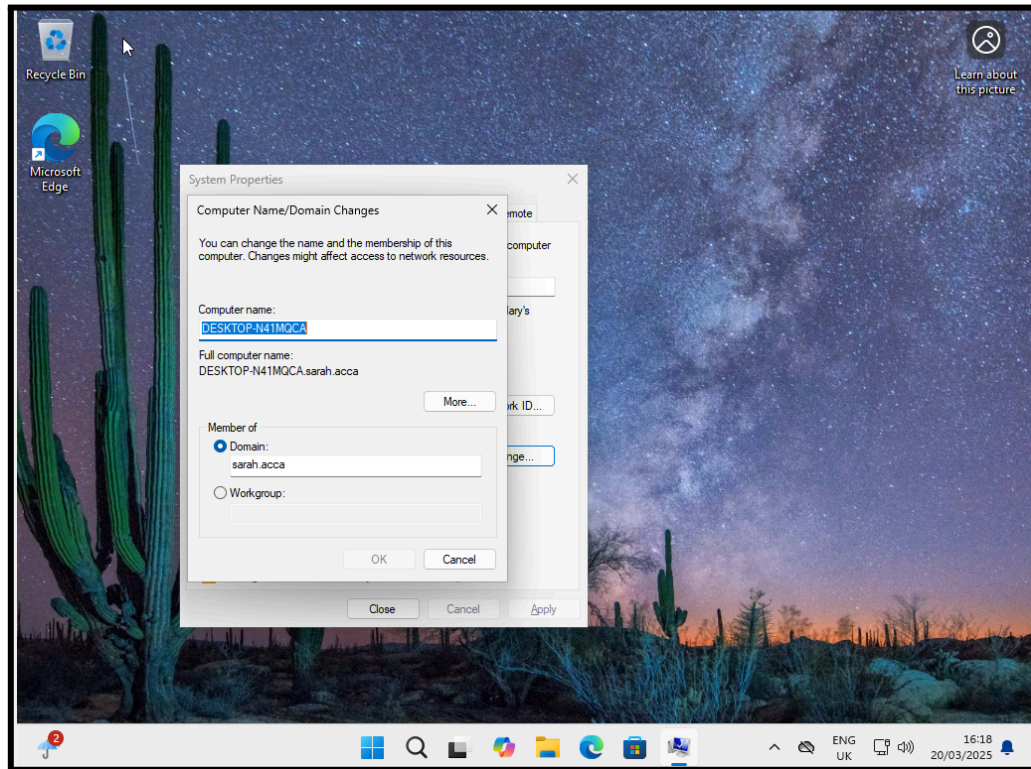
To validate the domain's functionality, a user account named **Sarah Levy** (**s.levy@sarah.acca**) was created using the Active Directory Users and Computers (ADUC) console.



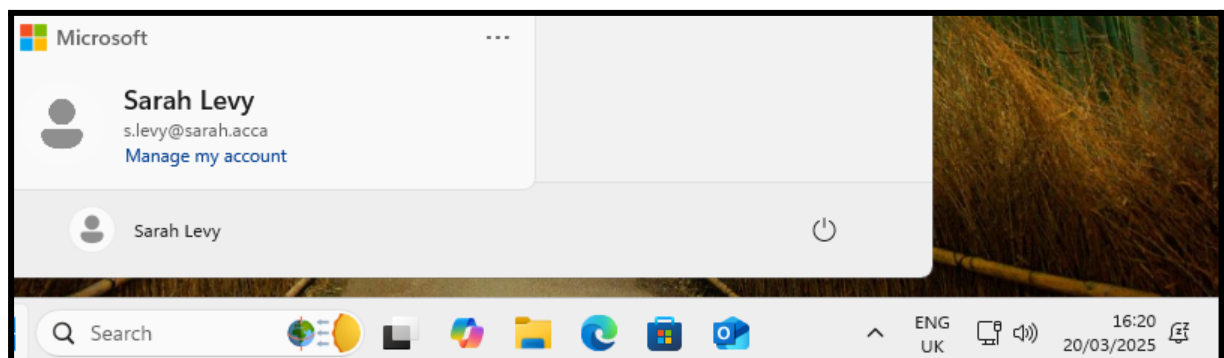
[18] Creating Sarah Levy's user

E. Joining a Client to the Domain

A windows 11 client was connected to the Swansea LAN and obtained an IP address via DHCP. The machine was then joined to the domain and tested by logging in with the *s.levy* account.



[19] Windows client joining the domain sarah.acca



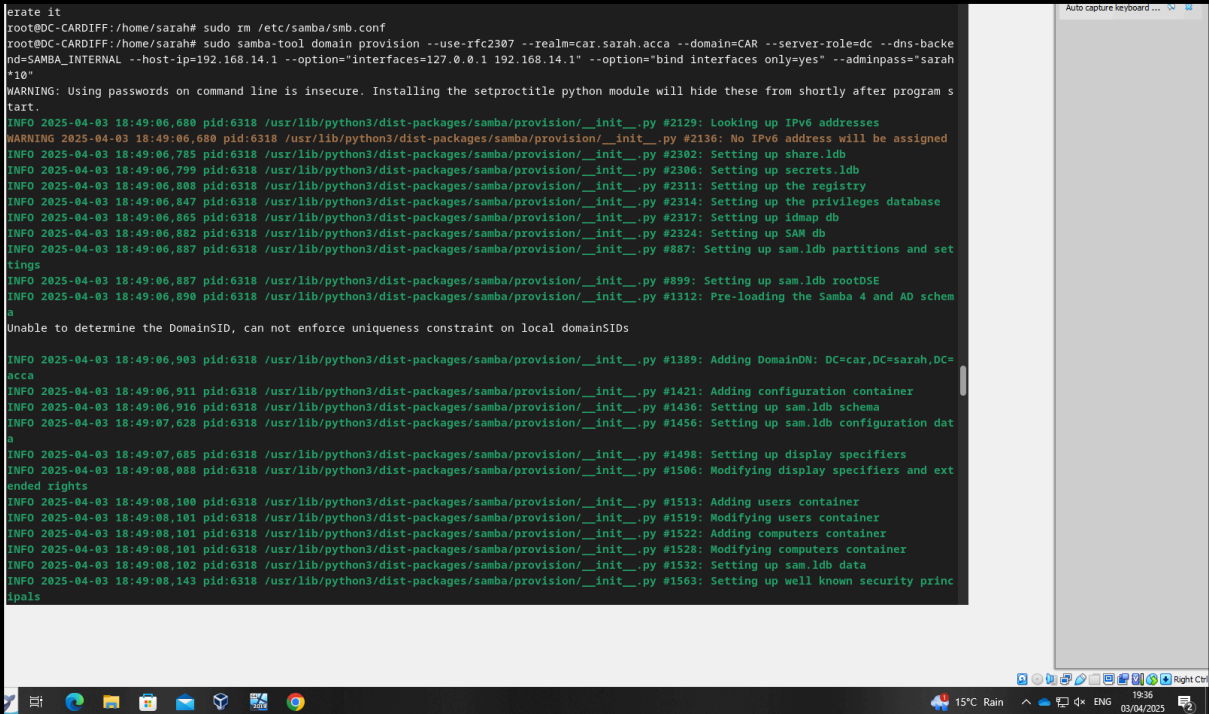
[20] Domain user logged in on the Windows client

2. Cardiff Child Domain

The Cardiff network was integrated into the main Active Directory structure by configuring a child domain named **car.sarah.acca**. The domain controller role was assigned to a Debian server using Samba and Kerberos. Once the child domain was provisioned, a domain trust was established with the parent domain in Swansea.

A. Provisioning the Debian Server as a Domain Controller

The Debian server on the Cardiff LAN was prepared with required packages: **samba**, **winbind**, **krb5-user**, and others. The **samba-tool domain provision** command was used to set up the domain **car.sarah.acca** as a child of **sarah.acca**.



```
create it
root@BDC-CARDIFF:/home/sarah# sudo rm /etc/samba/smb.conf
root@BDC-CARDIFF:/home/sarah# sudo samba-tool domain provision --use-irc2307 --realm=car.sarah.acca --domain=CAR --server-role=dc --dns-backend=SAMBA_INTERNAL --host-ip=192.168.14.1 --option="interfaces=127.0.0.1 192.168.14.1" --option="bind interfaces only=yes" --adminpass="sarah"10"
WARNING: Using passwords on command line is insecure. Installing the setproctitle python module will hide these from shortly after program starts.
INFO 2025-04-03 18:49:06,680 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2129: Looking up IPv6 addresses
WARNING 2025-04-03 18:49:06,680 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2136: No IPv6 address will be assigned
INFO 2025-04-03 18:49:06,785 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2302: Setting up share.ldb
INFO 2025-04-03 18:49:06,799 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2306: Setting up secrets.ldb
INFO 2025-04-03 18:49:06,808 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2311: Setting up the registry
INFO 2025-04-03 18:49:06,847 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2314: Setting up the privileges database
INFO 2025-04-03 18:49:06,865 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2317: Setting up idmap db
INFO 2025-04-03 18:49:06,882 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #2324: Setting up SAM db
INFO 2025-04-03 18:49:06,887 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #887: Setting up sam.ldb partitions and settings
INFO 2025-04-03 18:49:06,887 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #899: Setting up sam.ldb rootDSE
INFO 2025-04-03 18:49:06,890 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1312: Pre-loading the Samba 4 and AD schema
Unable to determine the DomainSID, can not enforce uniqueness constraint on local domainSIDs
INFO 2025-04-03 18:49:06,903 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1389: Adding DomainDN: DC=car,DC=sarah,DC=acca
INFO 2025-04-03 18:49:06,911 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1421: Adding configuration container
INFO 2025-04-03 18:49:06,916 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1436: Setting up sam.ldb schema
INFO 2025-04-03 18:49:07,628 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1456: Setting up sam.ldb configuration data
INFO 2025-04-03 18:49:07,685 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1498: Setting up display specifiers
INFO 2025-04-03 18:49:08,088 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1506: Modifying display specifiers and extended rights
INFO 2025-04-03 18:49:08,100 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1513: Adding users container
INFO 2025-04-03 18:49:08,101 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1519: Modifying users container
INFO 2025-04-03 18:49:08,101 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1522: Adding computers container
INFO 2025-04-03 18:49:08,101 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1528: Modifying computers container
INFO 2025-04-03 18:49:08,102 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1532: Setting up sam.ldb data
INFO 2025-04-03 18:49:08,143 pid:6318 /usr/lib/python3/dist-packages/samba/provision/_init_.py #1563: Setting up well known security principals
```

[21] Provisioning the Samba domain

```
GNU nano 2.2 /etc/samba/smb.conf
[global parameters]
[global]
interfaces = 127.0.0.1 192.168.14.1
bind interfaces only = yes
dns forwarder = 192.168.135.24
netbios name = DC-CARDIFF
dns forwarder = 10.14.27.1
realm = CAR.SARAH.ACCA
server role = active directory domain controller
workgroup = CAR
idmap_ldb:use rfc2307 = yes

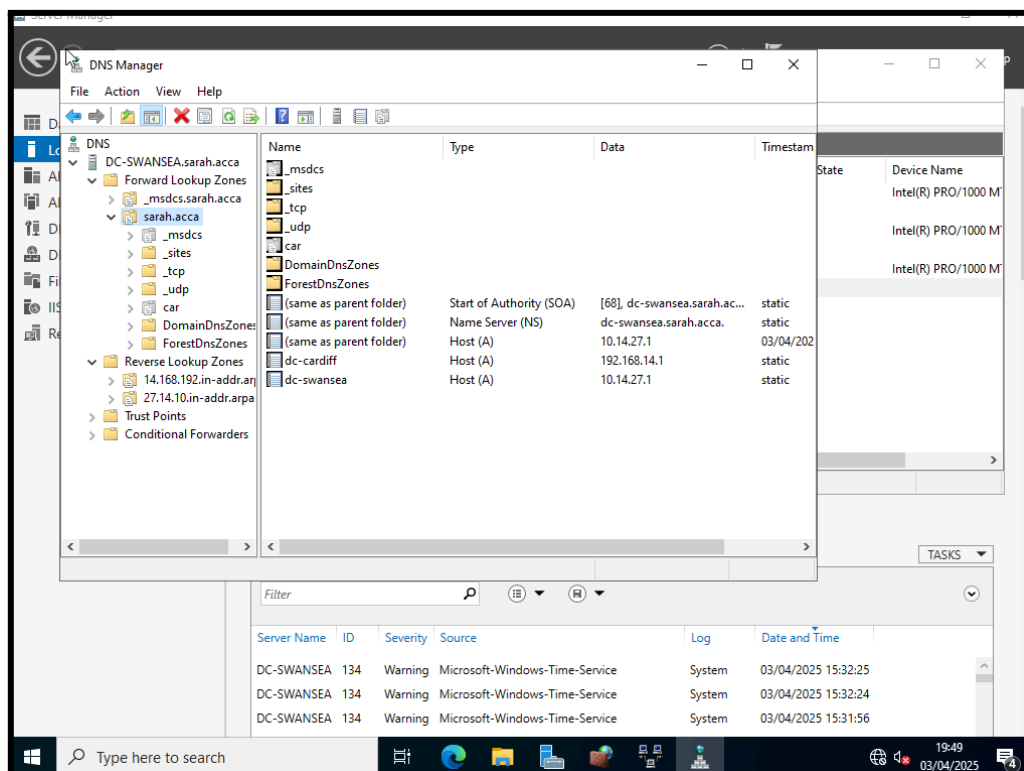
[sysvol]
path = /var/lib/samba/sysvol
read only = No

[netlogon]
path = /var/lib/samba/sysvol/car.sarah.acca/scripts
read only = No
```

[22] Final Samba configuration

B. DNS Integration with Parent Domain

A new host (A) record and corresponding PTR record were created on the DNS server in Swansea to allow name resolution of the Debian DC across domains as Windows Server will act as the main DNS Server for both networks.



[23] Final configuration of the DNS server

C. Adjusting Password Policy and Creating a Test User

The default Samba password policy was updated to disable complexity and minimum length requirements. A test user *s.levy@car.sarah.acca* was then created.

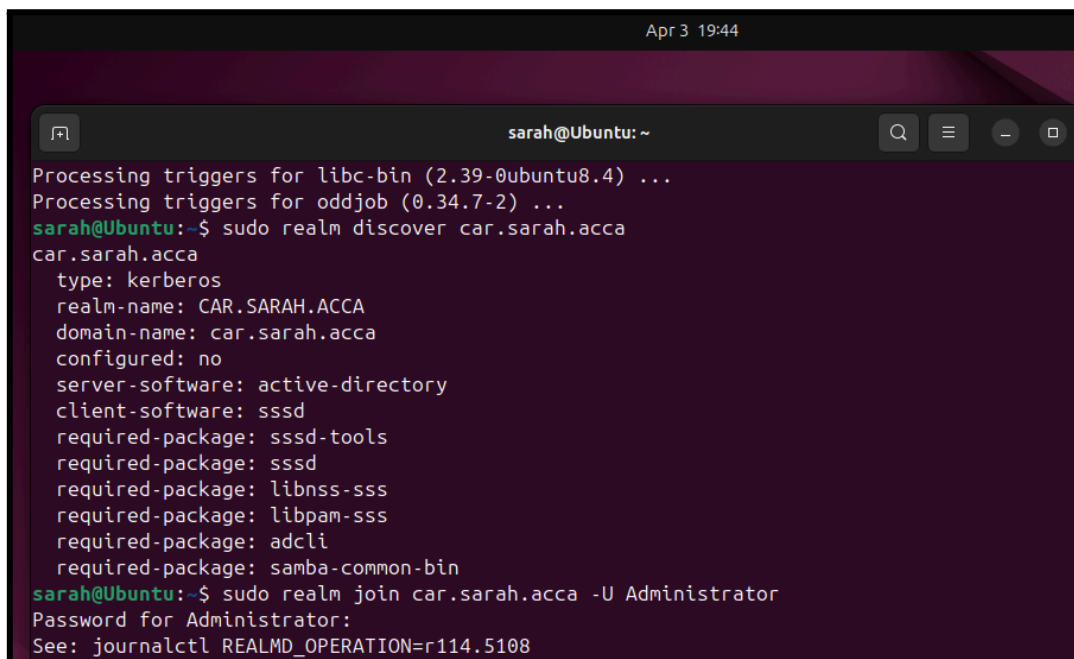
```
root@DC-CARDIFF:/home/sarah# sudo samba-tool domain passwordsettings set --complexity=off
Password complexity deactivated!
All changes applied successfully!
root@DC-CARDIFF:/home/sarah# sudo samba-tool domain passwordsettings set --min-pwd-length=0
Usage: samba-tool domain passwordsettings set <options> [options]

samba-tool domain passwordsettings set: error: no such option: --min-pwd-length
root@DC-CARDIFF:/home/sarah# sudo samba-tool domain passwordsettings set --min-pwd-length=0
Minimum password length changed!
All changes applied successfully!
root@DC-CARDIFF:/home/sarah# sudo samba-tool user create s.levy 'sarah' --given-name='Sarah' --surname='Levy'
User 's.levy' added successfully
```

[24] Successful password policy changes and user creation with samba-tool.

D. PAM and Realm Configuration on Ubuntu Client

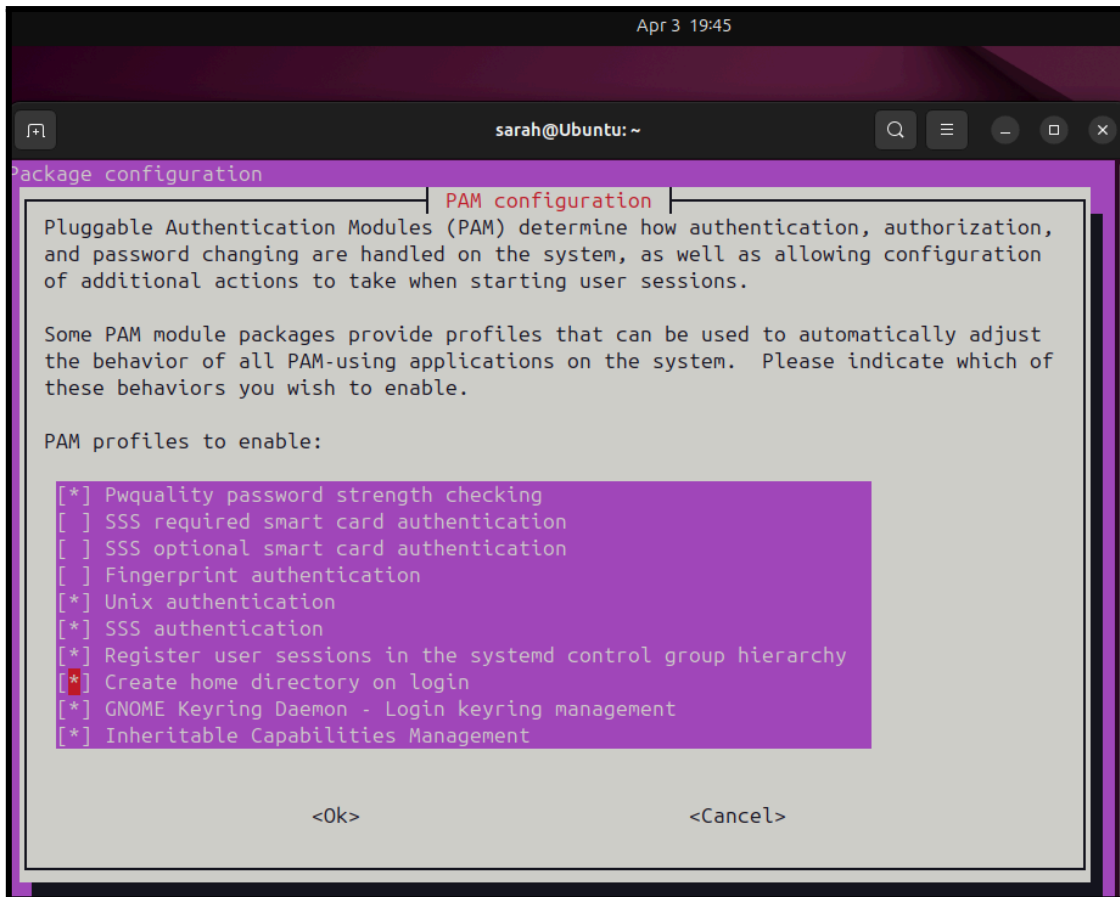
Kerberos integration is crucial for AD to function correctly. The Ubuntu client was joined to the child domain using the realm join command. This domain is discoverable by DNS thanks to the earlier configured new DNS records on Windows Server.



```
Apr 3 19:44
sarah@Ubuntu: ~
Processing triggers for libc-bin (2.39-0ubuntu8.4) ...
Processing triggers for oddjob (0.34.7-2) ...
sarah@Ubuntu:~$ sudo realm discover car.sarah.acca
car.sarah.acca
  type: kerberos
  realm-name: CAR.SARAH.ACCA
  domain-name: car.sarah.acca
  configured: no
  server-software: active-directory
  client-software: sssd
  required-package: sssd-tools
  required-package: sssd
  required-package: libnss-sss
  required-package: libpam-sss
  required-package: adcli
  required-package: samba-common-bin
sarah@Ubuntu:~$ sudo realm join car.sarah.acca -U Administrator
Password for Administrator:
See: journalctl REALMD_OPERATION=r114.5108
```

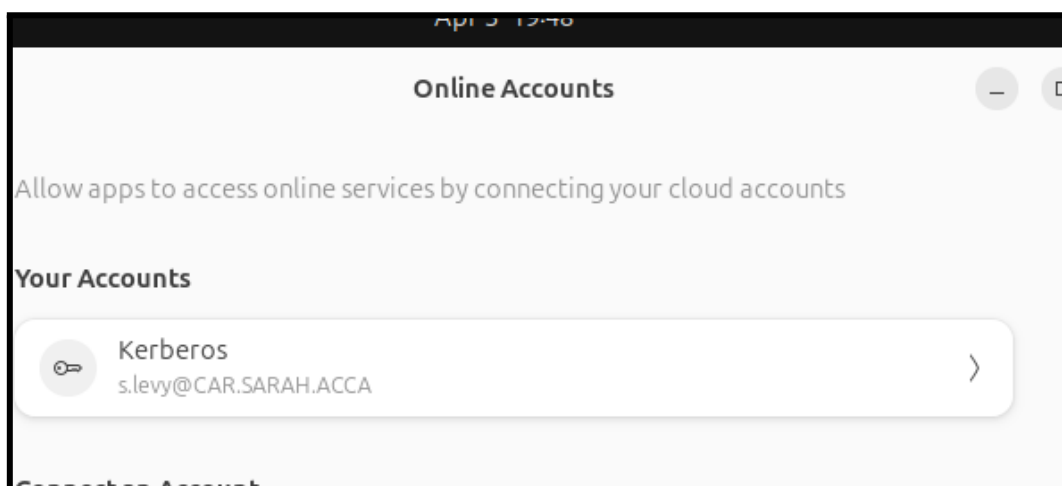
[27] Joining the child domain on the Ubuntu client

To allow domain logins on a Linux client, the *realmd*, *sssd* and PAM were configured. Home directories are automatically created for new users upon login.



[25] Home directory creation module enabled

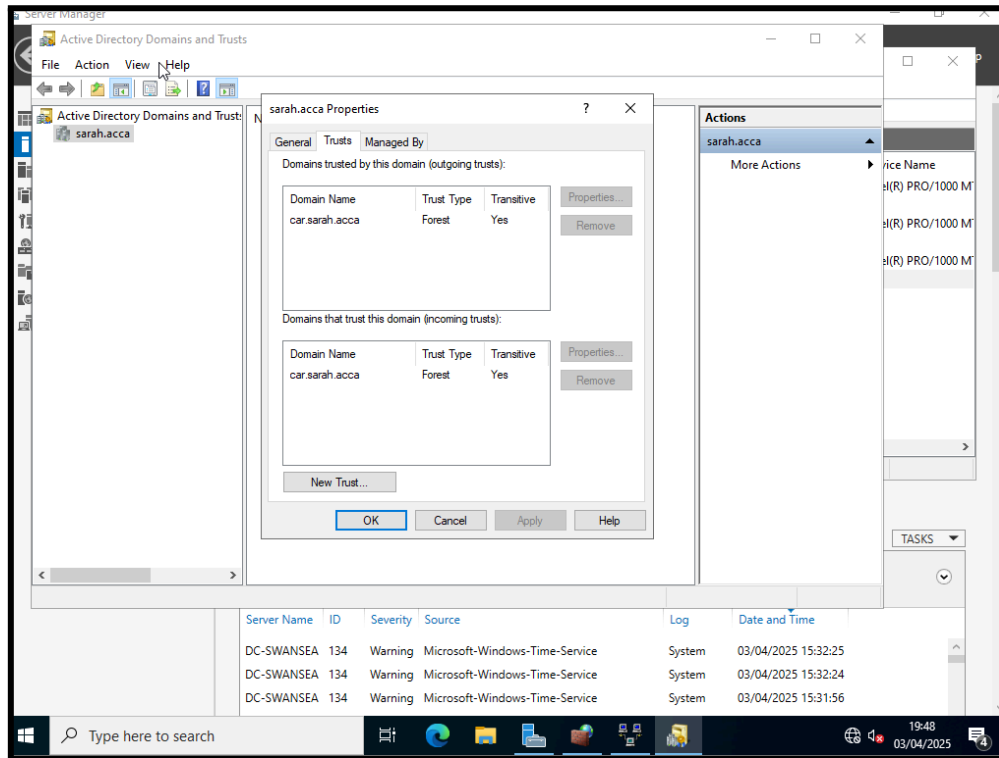
E. Verifying Kerberos Authentication and Domain Integration



[26] Ubuntu client successfully joined the car.sarah.acca domain

F. Establishing Trust Between Domains

A bidirectional, transitive forest trust was configured between **sarah.acca** and **car.sarah.acca**. This allows users and resources to be authenticated across both domains.



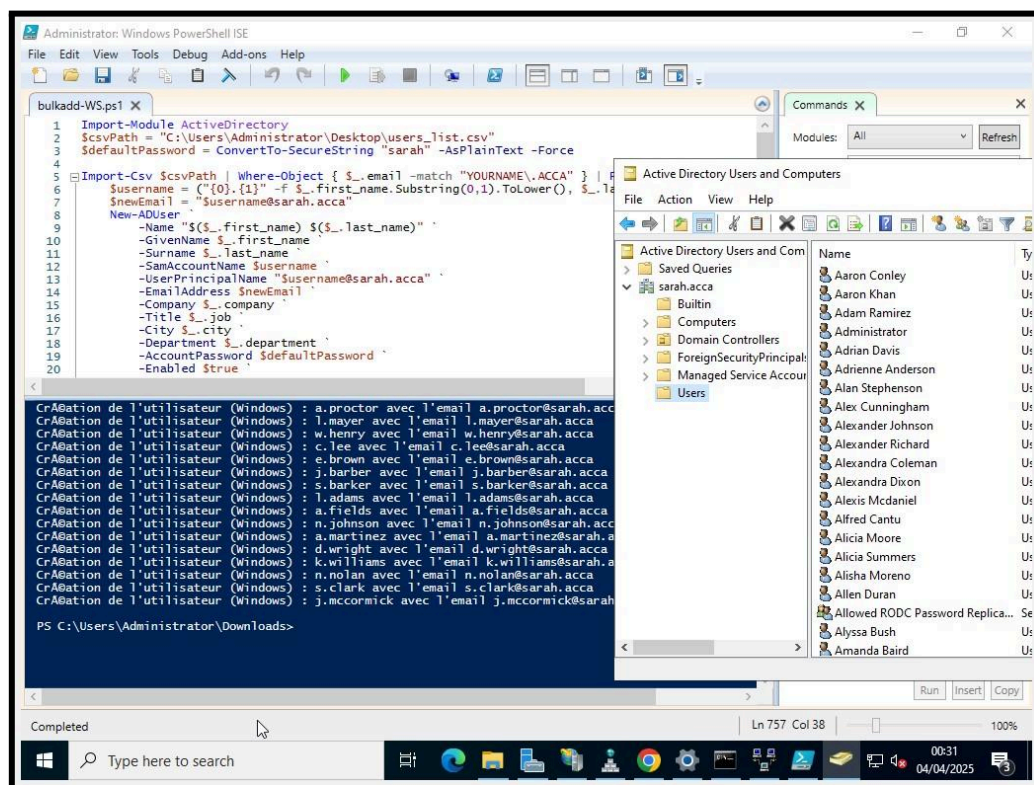
[28] Successful trust between parent and child domains

Importing Users

To create all the users needed for the network, the CSV file named *users_list_updated.csv* was downloaded from moodle. Users were added automatically using scripts, one for each domain. The users who had **YOURNAME.ACCA** as their domain were added on the Swansea server, and the users with **OLDNAME.ACCA** were added on the Cardiff server.

1. Users in the Swansea Domain

On the Windows Server in Swansea, a PowerShell script was used to read and import the users. The script created each user with their name, email address, job title, department, and other information from the CSV file. The format for the usernames was the first letter of the first name followed by the last name (for example j.doe).

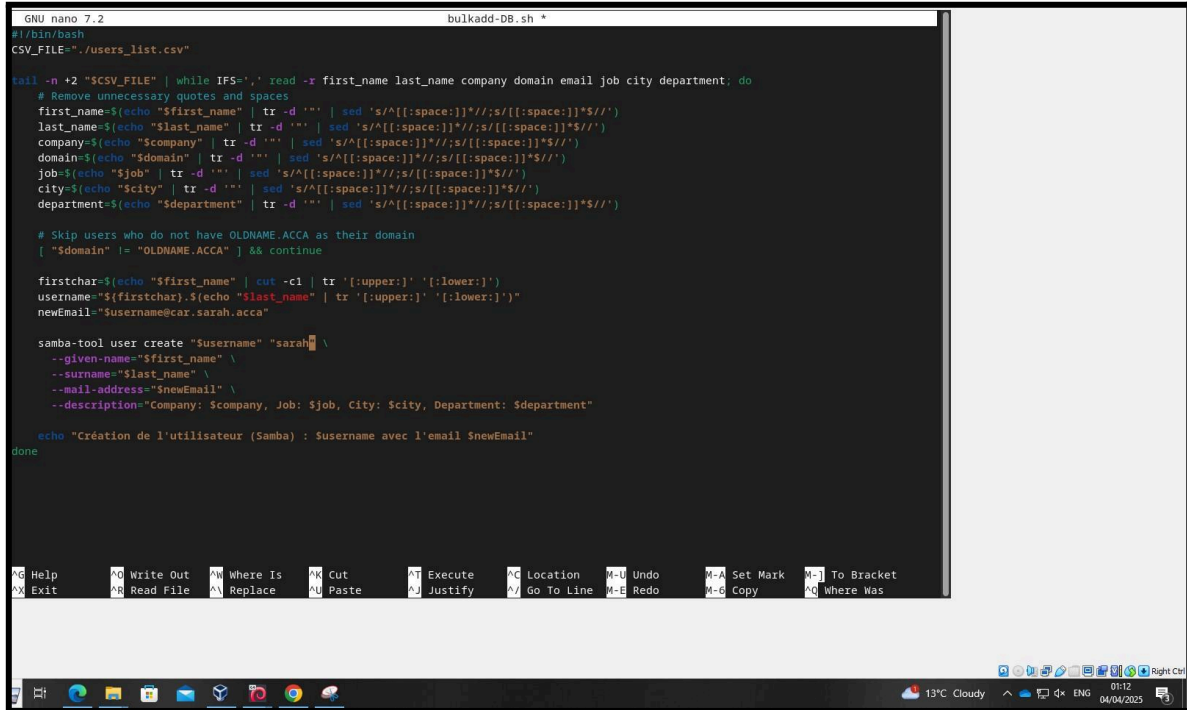


[29] The PowerShell script

Each user got the same default password and their account was active right away.

2. Users in the Cardiff Domain

On the Debian server in Cardiff, a Bash script was used together with the **samba-tool**. Unlike Windows, Samba cannot use fields like “company”, “job title” or “city” directly. So all of that information was written into the description field for each user.



```
GNU nano 7.2 bulkadd-DB.sh *
#!/bin/bash
CSV_FILE="/users_list.csv"

tail -n +2 "$CSV_FILE" | while IFS="," read -r first_name last_name company domain email job city department; do
    # Remove unnecessary quotes and spaces
    first_name=$(echo "$first_name" | tr -d '"' | sed 's/^[[:space:]]*//;s/[[:space:]]*$//')
    last_name=$(echo "$last_name" | tr -d '"' | sed 's/^[[:space:]]*//;s/[[:space:]]*$//')
    company=$(echo "$company" | tr -d '"' | sed 's/^[[:space:]]*//;s/[[:space:]]*$//')
    domain=$(echo "$domain" | tr -d '"' | sed 's/^[[:space:]]*//;s/[[:space:]]*$//')
    job=$(echo "$job" | tr -d '"' | sed 's/^[[:space:]]*//;s/[[:space:]]*$//')
    city=$(echo "$city" | tr -d '"' | sed 's/^[[:space:]]*//;s/[[:space:]]*$//')
    department=$(echo "$department" | tr -d '"' | sed 's/^[[:space:]]*//;s/[[:space:]]*$//')

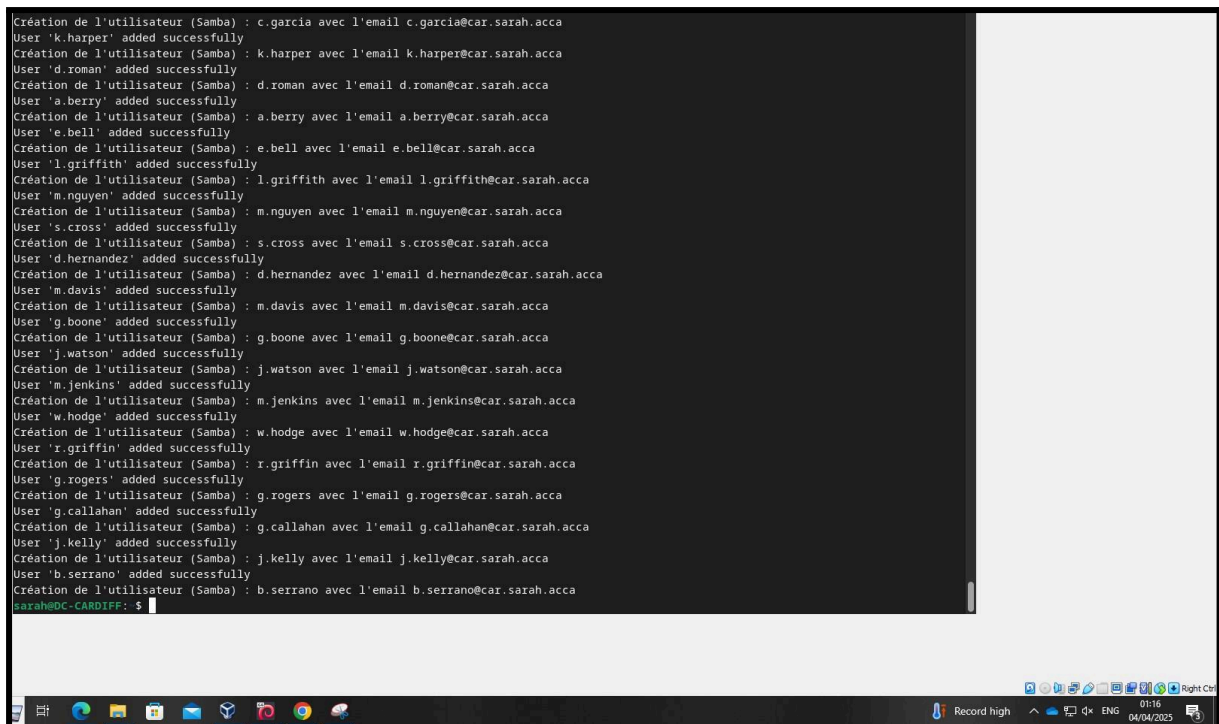
    # Skip users who do not have OLDNAME.ACCA as their domain
    [ "$domain" != "OLDNAME.ACCA" ] && continue

    firstchar=$(echo "$first_name" | cut -c1 | tr '[:upper:]' '[:lower:]')
    username="$firstchar.${last_name,,}"
    newEmail="$username@car.sarah.acca"

    samba-tool user create "$username" "sarah" \
        --given-name="$first_name" \
        --surname="$last_name" \
        --mail-address="$newEmail" \
        --description="Company: $company, Job: $job, City: $city, Department: $department"

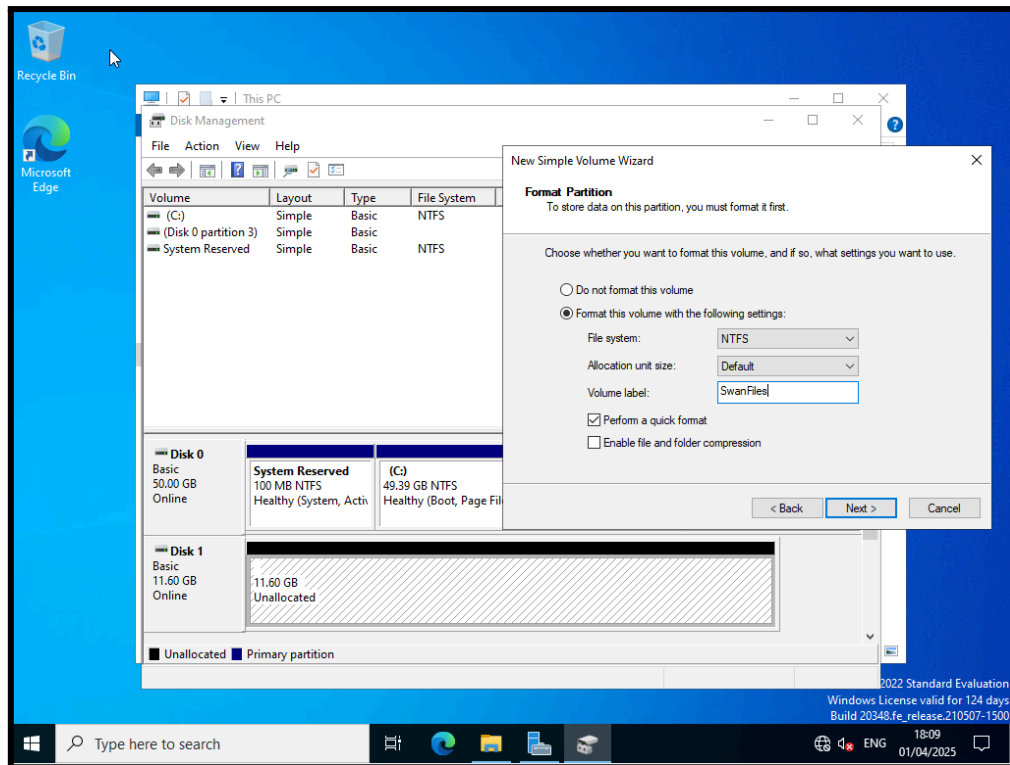
    echo "Création de l'utilisateur (Samba) : $username avec l'email $newEmail"
done
```

[30] Script debian to add users

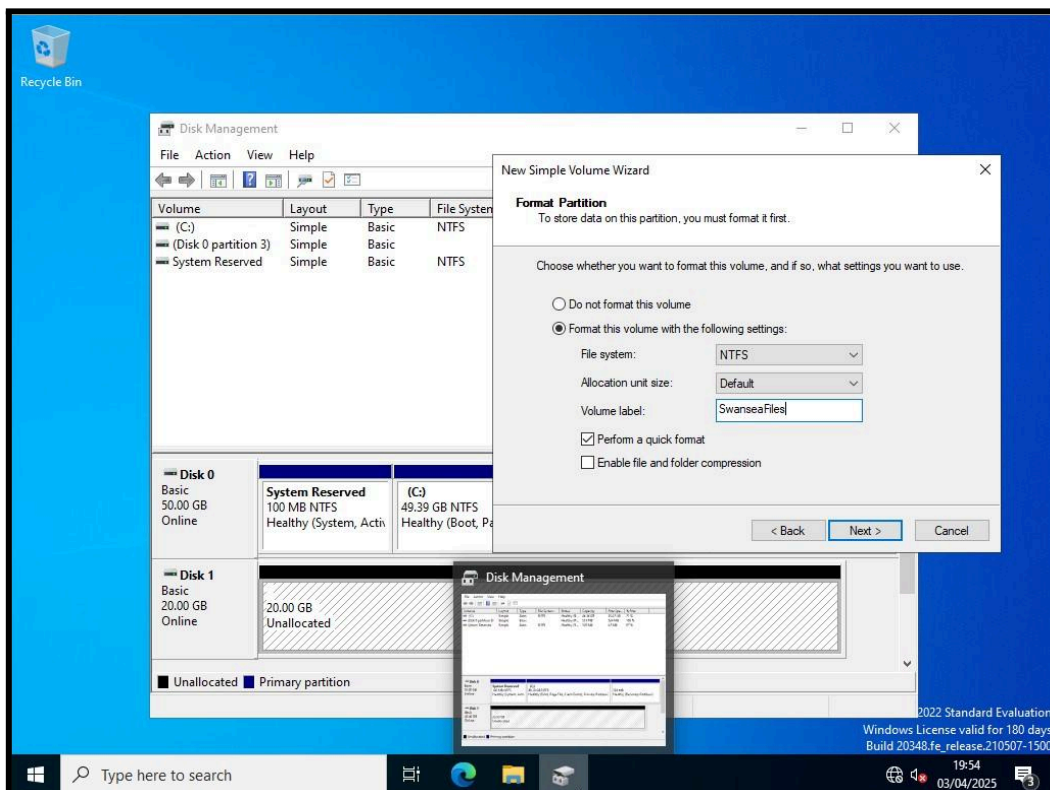


```
Création de l'utilisateur (Samba) : c.garcia avec l'email c.garcia@car.sarah.acca
User 'k.harper' added successfully
Création de l'utilisateur (Samba) : k.harper avec l'email k.harper@car.sarah.acca
User 'd.roman' added successfully
Création de l'utilisateur (Samba) : d.roman avec l'email d.roman@car.sarah.acca
User 'a.berry' added successfully
Création de l'utilisateur (Samba) : a.berry avec l'email a.berry@car.sarah.acca
User 'e.bell' added successfully
Création de l'utilisateur (Samba) : e.bell avec l'email e.bell@car.sarah.acca
User 'l.griffith' added successfully
Création de l'utilisateur (Samba) : l.griffith avec l'email l.griffith@car.sarah.acca
User 'm.nguyen' added successfully
Création de l'utilisateur (Samba) : m.nguyen avec l'email m.nguyen@car.sarah.acca
User 's.cross' added successfully
Création de l'utilisateur (Samba) : s.cross avec l'email s.cross@car.sarah.acca
User 'd.hernandez' added successfully
Création de l'utilisateur (Samba) : d.hernandez avec l'email d.hernandez@car.sarah.acca
User 'm.davis' added successfully
Création de l'utilisateur (Samba) : m.davis avec l'email m.davis@car.sarah.acca
User 'g.boone' added successfully
Création de l'utilisateur (Samba) : g.boone avec l'email g.boone@car.sarah.acca
User 'j.watson' added successfully
Création de l'utilisateur (Samba) : j.watson avec l'email j.watson@car.sarah.acca
User 'm.jenkins' added successfully
Création de l'utilisateur (Samba) : m.jenkins avec l'email m.jenkins@car.sarah.acca
User 'w.hodge' added successfully
Création de l'utilisateur (Samba) : w.hodge avec l'email w.hodge@car.sarah.acca
User 'r.griffin' added successfully
Création de l'utilisateur (Samba) : r.griffin avec l'email r.griffin@car.sarah.acca
User 'g.rogers' added successfully
Création de l'utilisateur (Samba) : g.rogers avec l'email g.rogers@car.sarah.acca
User 'g.callahan' added successfully
Création de l'utilisateur (Samba) : g.callahan avec l'email g.callahan@car.sarah.acca
User 'j.kelly' added successfully
Création de l'utilisateur (Samba) : j.kelly avec l'email j.kelly@car.sarah.acca
User 'b.serrano' added successfully
Création de l'utilisateur (Samba) : b.serrano avec l'email b.serrano@car.sarah.acca
sarah@dc-CARDIFF: $
```

[31] Creating users on Debian



[33] Formatting a disk and assigning the label "SwanseaFiles"



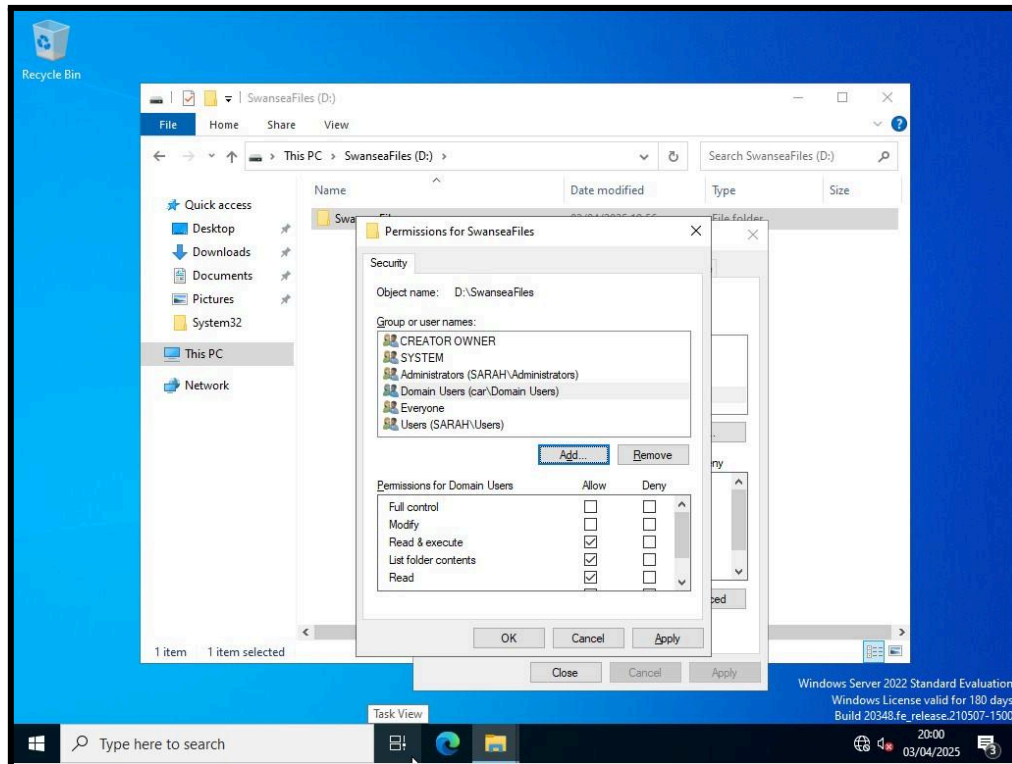
[34] Formatting a disk with the NTFS file system and label "SwanFiles"

Once formatted, the disk was mounted as a new volume, and a folder named SwanseaFiles was created inside.

2. Sharing and Configuring permissions

The folder was shared and security permissions were adjusted to reflect access needs:

- Full control was granted to users from the parent domain **sarah.acca**
- Read & Execute rights were granted to users from the child domain **car.sarah.acca**
- **Everyone** and **SARAH\Users** were included to avoid unexpected access issues



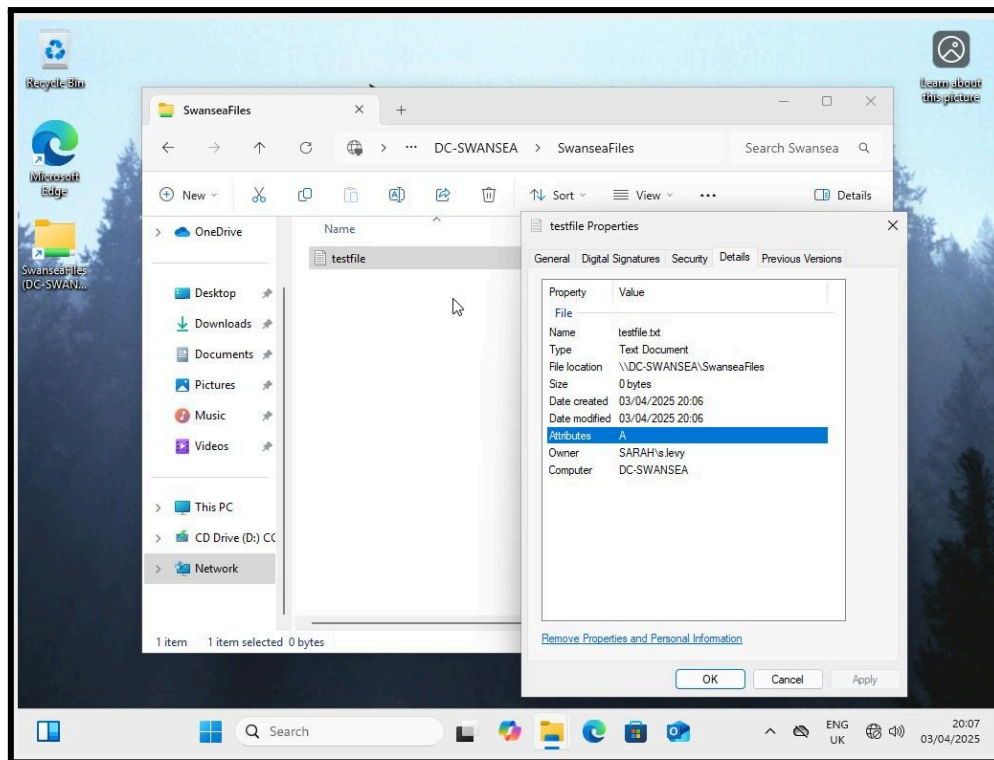
[35] Configuring shared folder permissions for domains users from both Swansea and Cardiff

This ensured that Swansea users could create, edit, and delete files, while Cardiff users had read-only access.

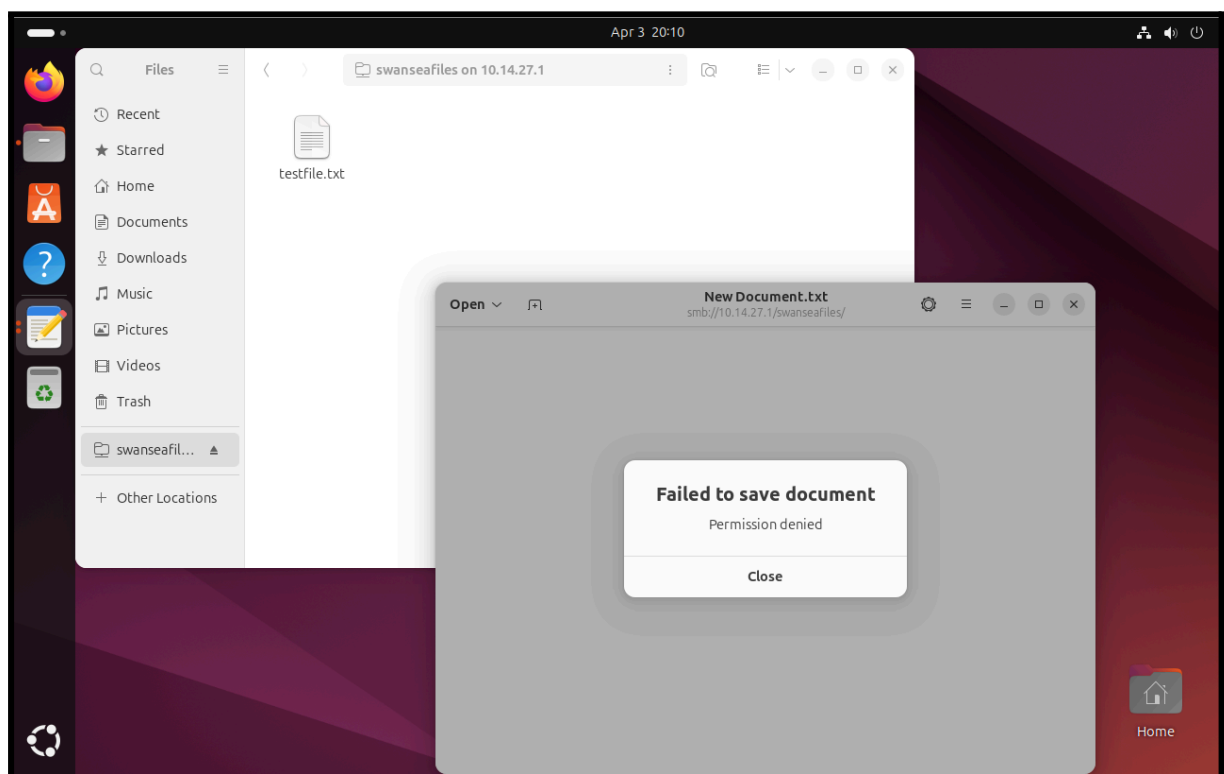
3. Testing Access Rights

Tests were run using domain users from both locations to check that access control worked as expected.

- NETWORK OPERATING SYSTEMS AND SERVICES -



[36] Swansea domain user successfully creates a file on the shared folder



[35] Cardiff domain user is denied write access when trying to create a file on the share

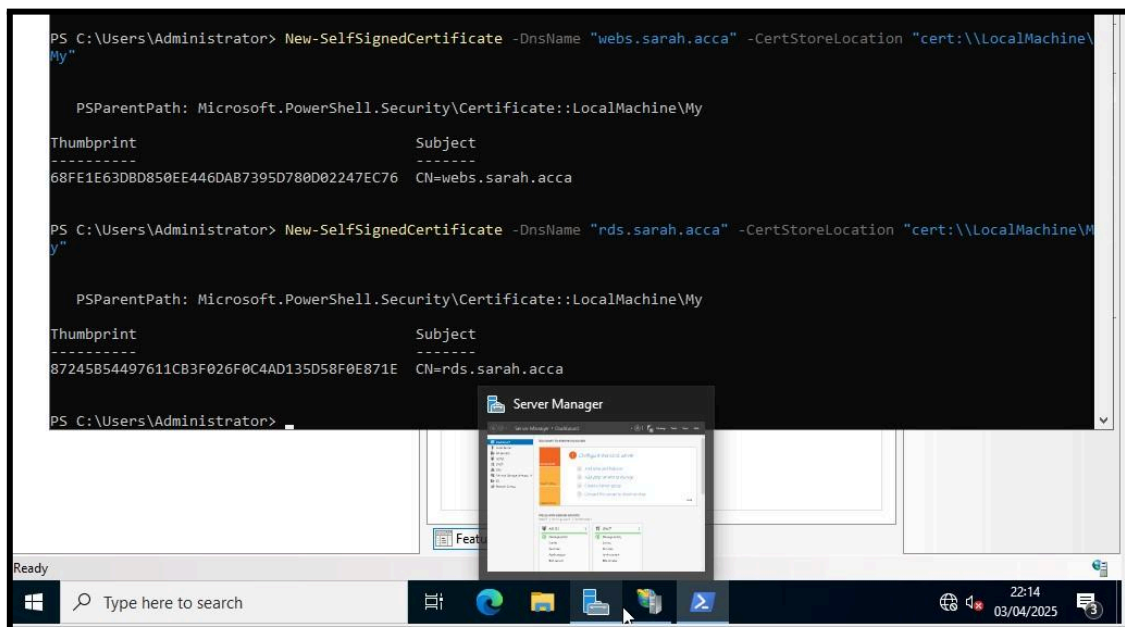
The test confirms that permissions were correctly set according to the user's domain.

ISS Web Server Setup

To publish a simple internal website for testing, the Internet Information Services (ISS) role was added to the Windows Server in Swansea. A custom HTML file was then served under a custom domain name using HTTPS with a self-signed certificate.

1. Generating a Self-Signed SSL Certificate

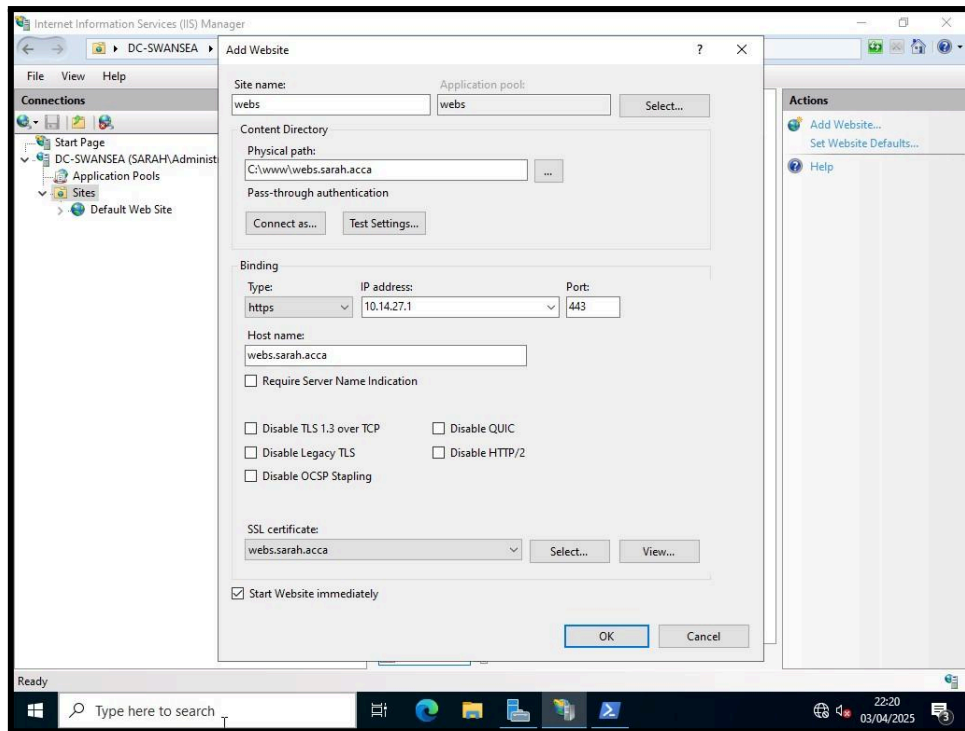
A self-signed certificate was created using PowerShell to allow HTTPS access to the website. The certificate was bound to the domain ***webs.sarah.acca***.



[37] Creating self-signed certificates for webs.sarah.acca and rds.sarah.acca with PowerShell

2. Setting up the Website in IIS

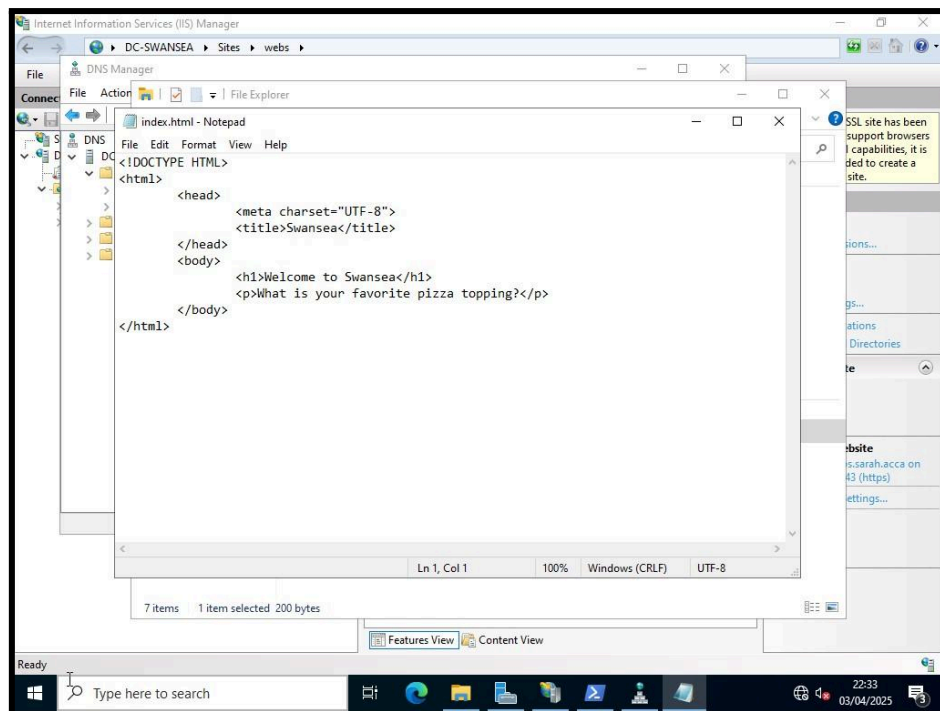
In the IIS Manager, a new website named ***webs*** was created. The root folder was set to ***C:\www\webs.sarah.acca***, and the site was configured to listen on ***IP 10.14.27.1***, port 443, using the HTTPS protocol. The hostname ***webs.sarah.acca*** was used to identify the website.



[38] IIS configuration for the new website using HTTPS binding

3. Creating the HTML Page

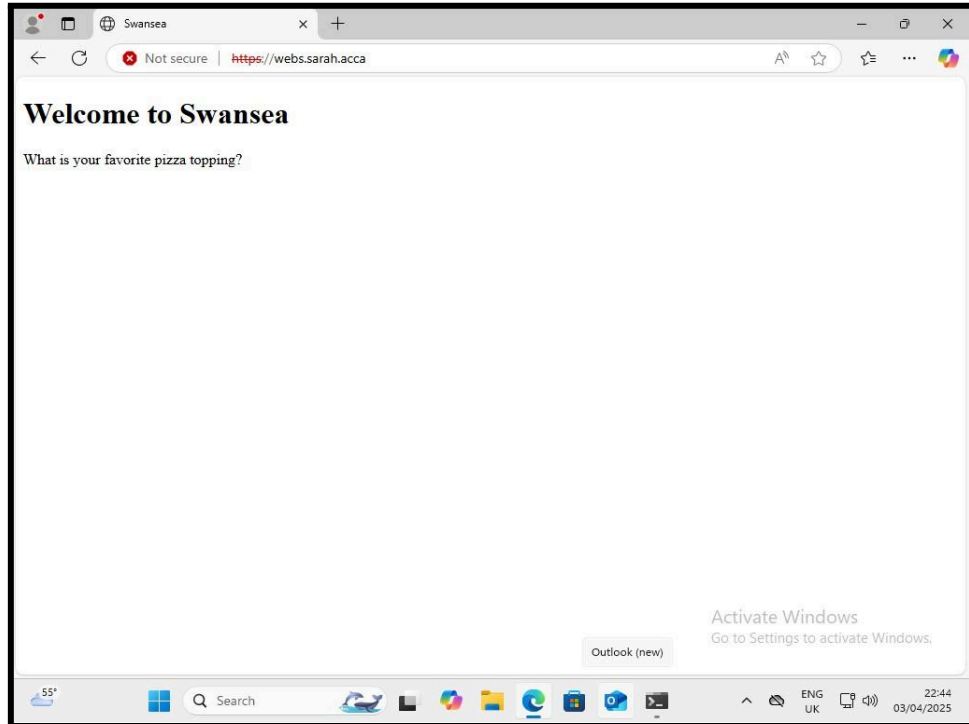
A very basic HTML page was created manually using Notepad and saved as *index.html*.



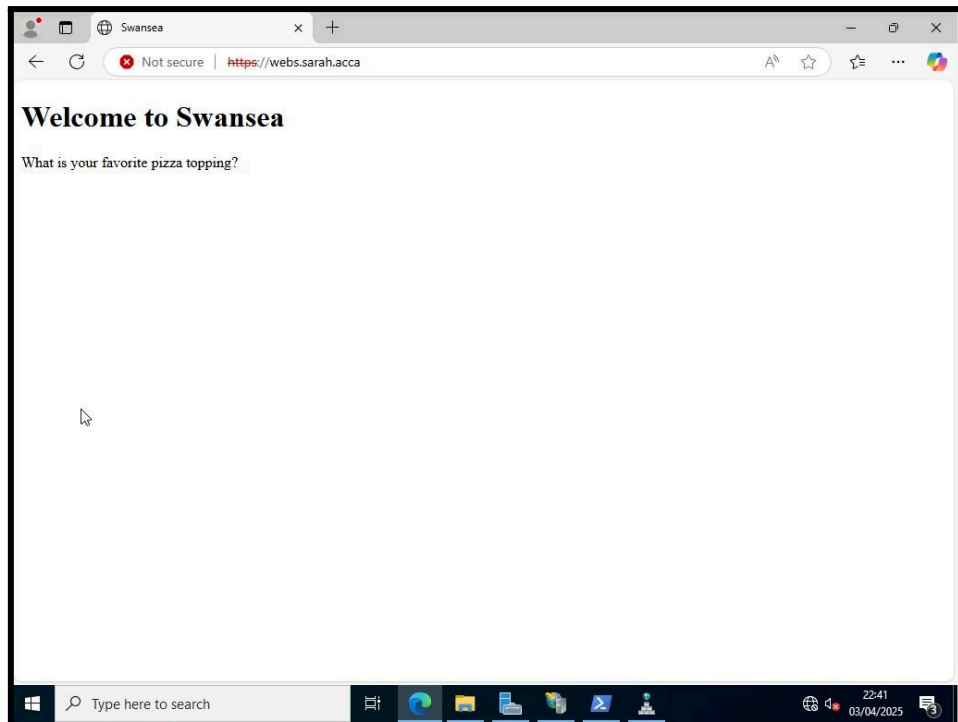
[39] HTML code written for the test website

4. Verifying the Website on Clients

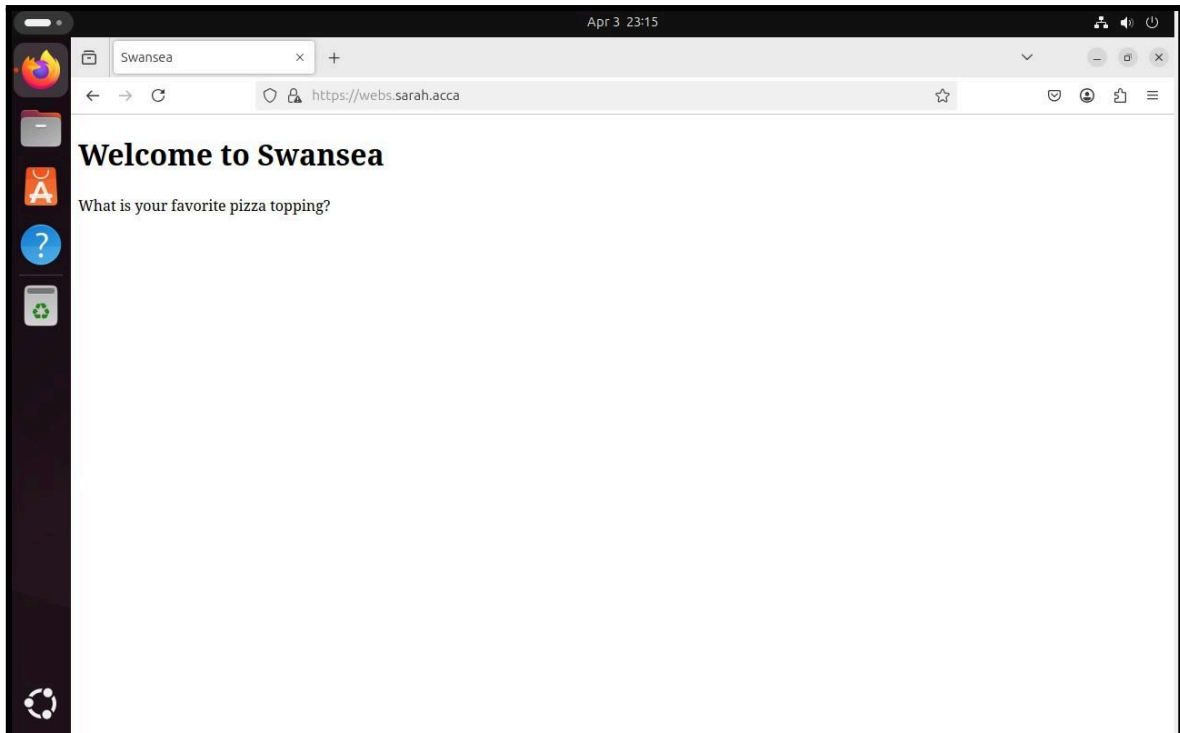
The website was tested from several machines. The domain ***webs.sarah.acca*** was correctly resolved, and the page loaded with the expected message.



[40] The Swansea website loaded successfully on Windows 11 client.

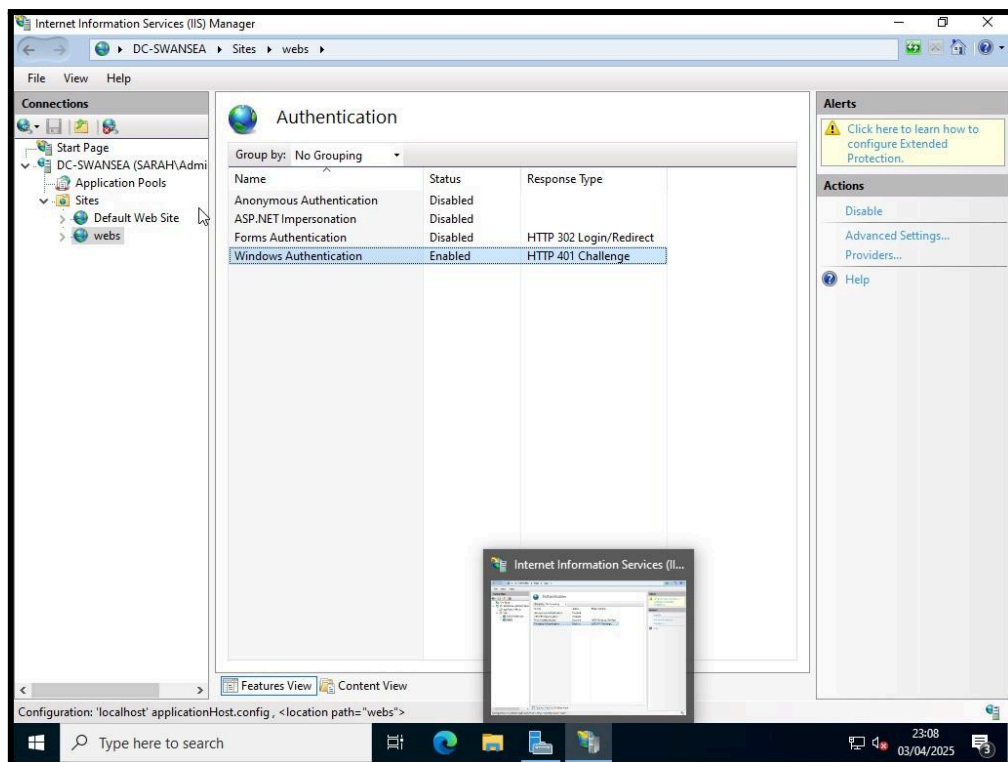


[41] The Swansea website loaded successfully on Windows Server 2022



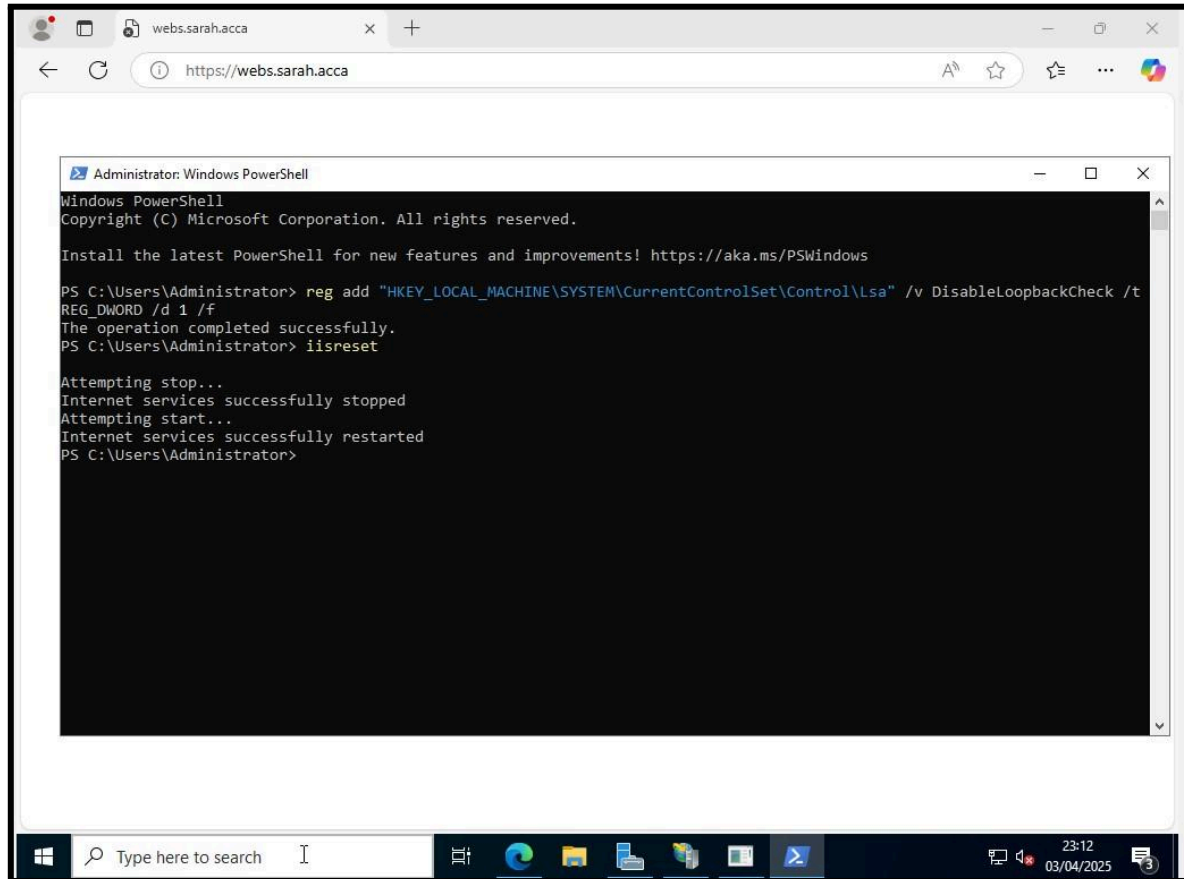
[42] The Swansea website loaded successfully on Ubuntu Client

Afterwards, we activated the windows authentication for this website.



[43] Enabling Windows Authentication

The issue with enabling windows authentication is that it wasn't working right out of the box with windows server 2022, so the access to the website from it was no longer available afterwards as it wasn't possible to login to any accounts. A solution was found by disabling the loopback check with a registry value using PowerShell.



[44] Disabling the loopback check rule with PowerShell

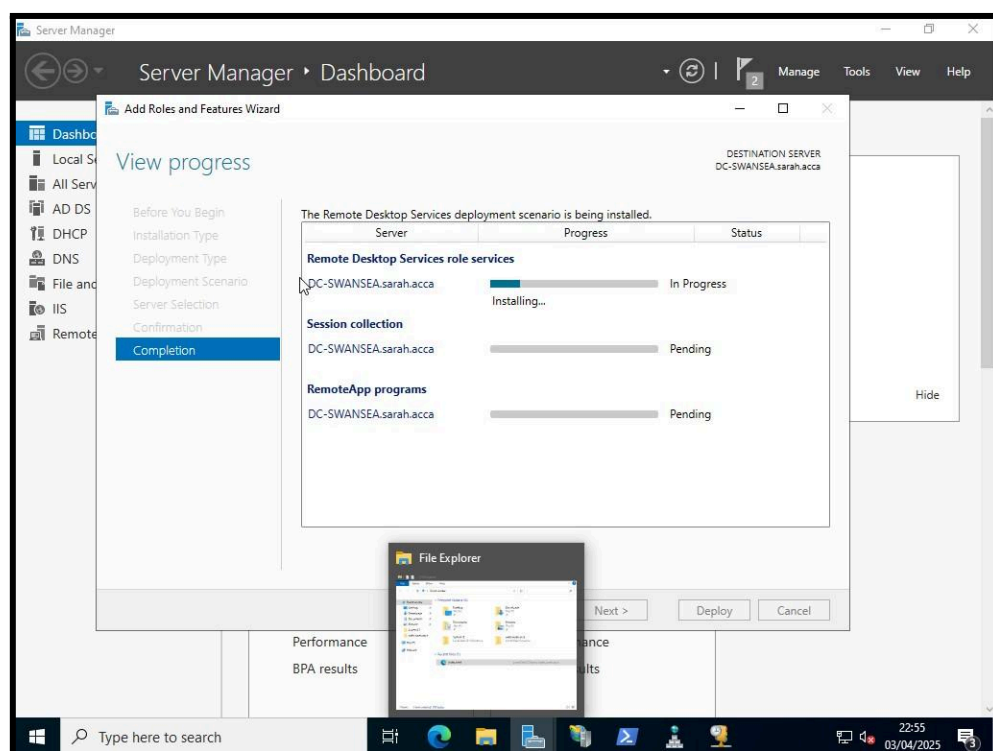
Thus the website became available on Windows Server with a working authentication.

Remote Desktop Services (RDS)

Remote Desktop Services were deployed on the DC-SWANSEA server to allow users from both Swansea and Cardiff domains to access Windows applications remotely via web browser. The installation includes role setup, session collection configuration, application publishing, and user access testing.

1. Installing the RDS Role

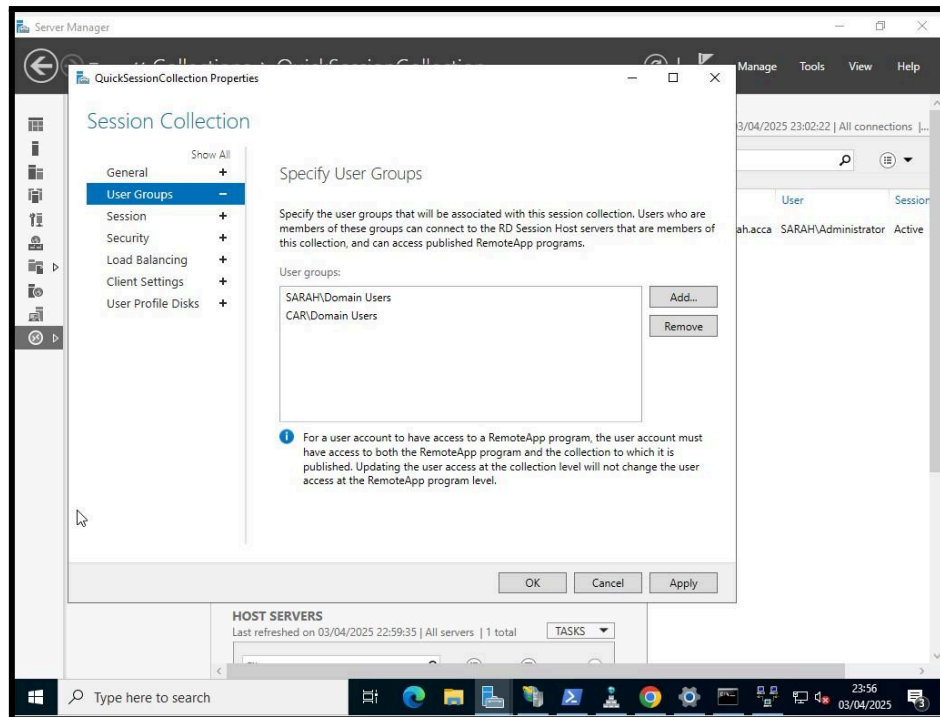
The setup began by adding the Remote Desktop Services role via the Server Manager wizard. The scenario included RD Session Host, RD Web Access, and a basic session collection.



[45] Installing the Remote Desktop Services role on DC-SWANSEA

2. Creating the Session Collection and Assigning Users

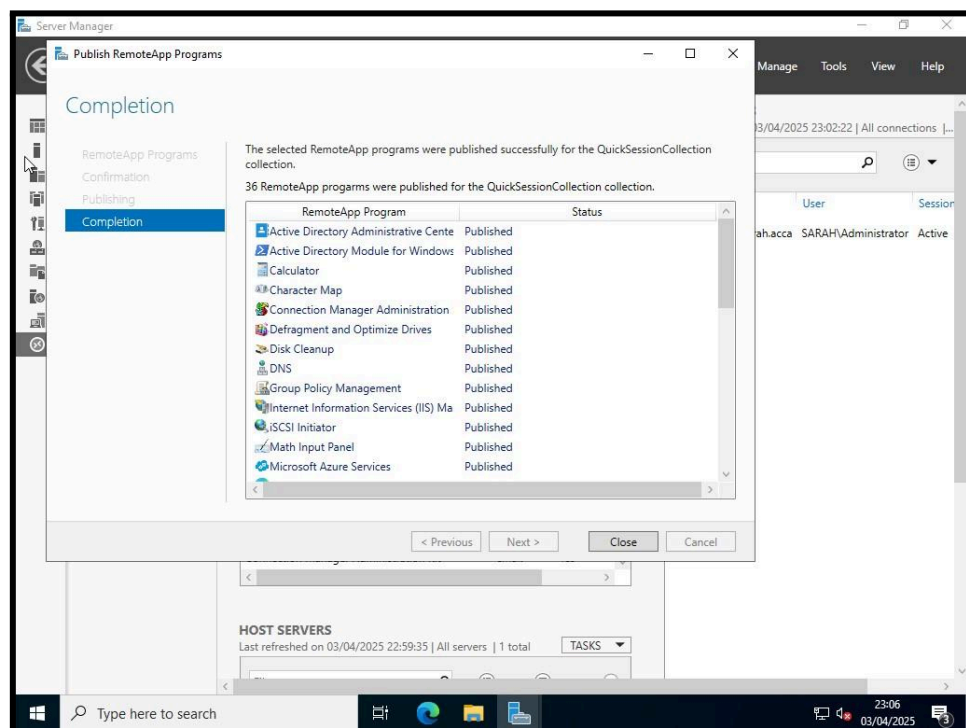
A session collection was automatically created and named QuickSessionCollection. Both **SARAH\Domain Users** and **CARL\Domain Users** were added to the user groups so that users from both domains could connect.



[46] Adding both Swansea and Cardiff domain users to the RDS session collection

3. Publishing Remote Applications

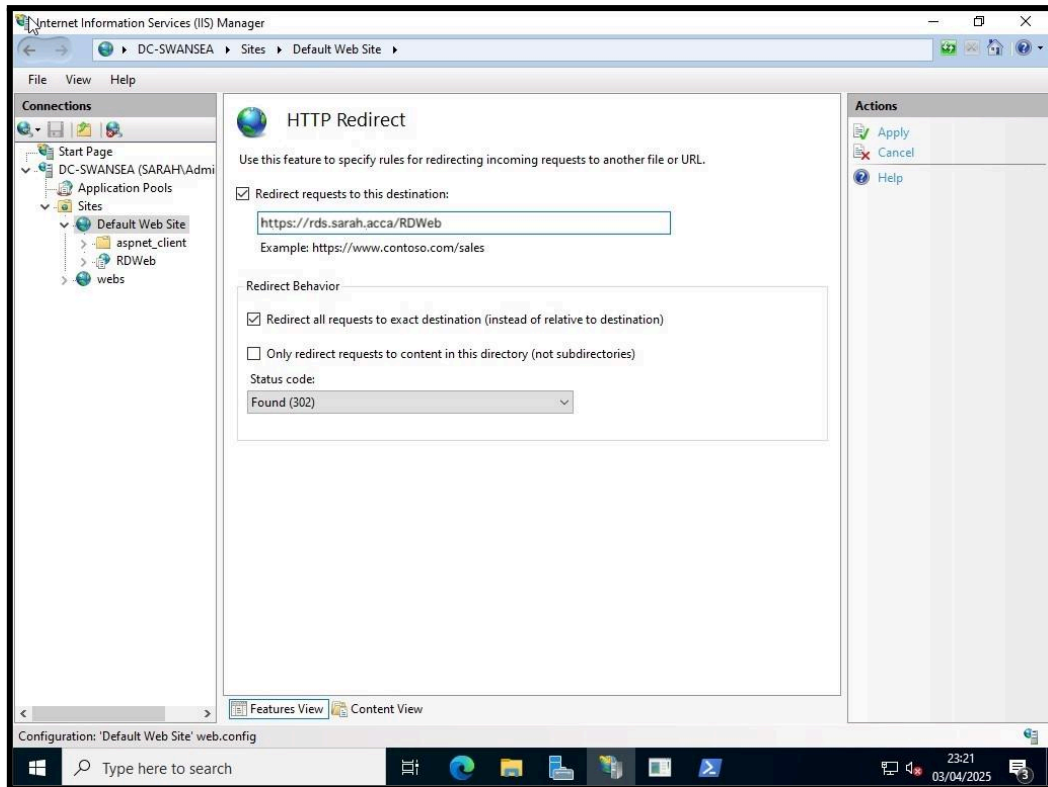
A selection of applications was made available to users through RD Web Access. These included essential tools like Active Directory, Group Policy Management, Paint, and Calculator.



[47] RemoteApp programs published in QuickSessionCollection

4. Configuring IIS Redirect

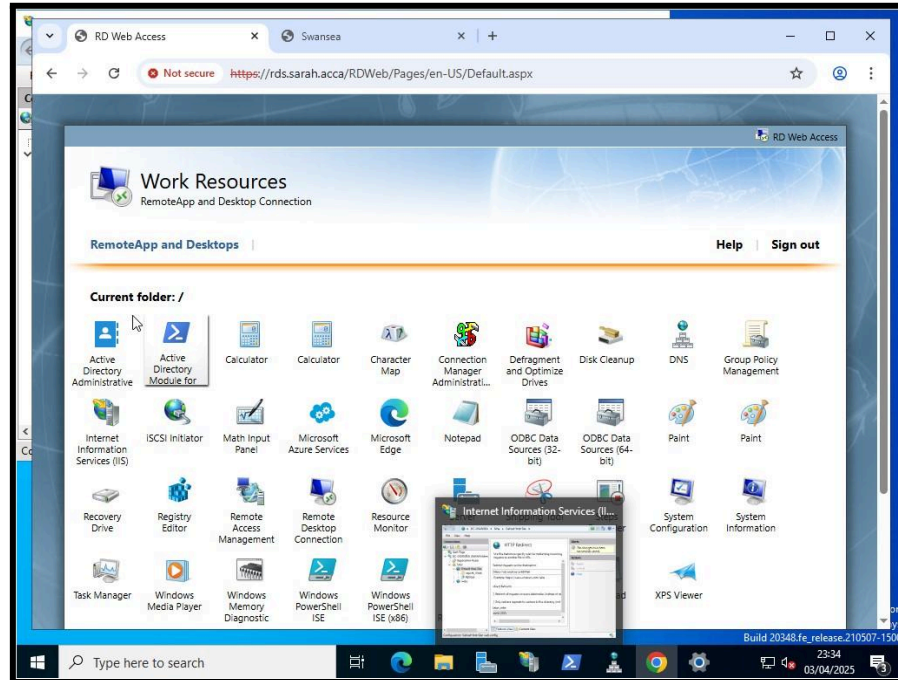
To make access easier, a redirect rule was added to the Default Web Site in IIS. This rule forwards any request to the main hostname. (<https://rds.sarah.acca>) directly to the RD Web portal.



[48] HTTP redirect from the main site to RD Web portal in IIS

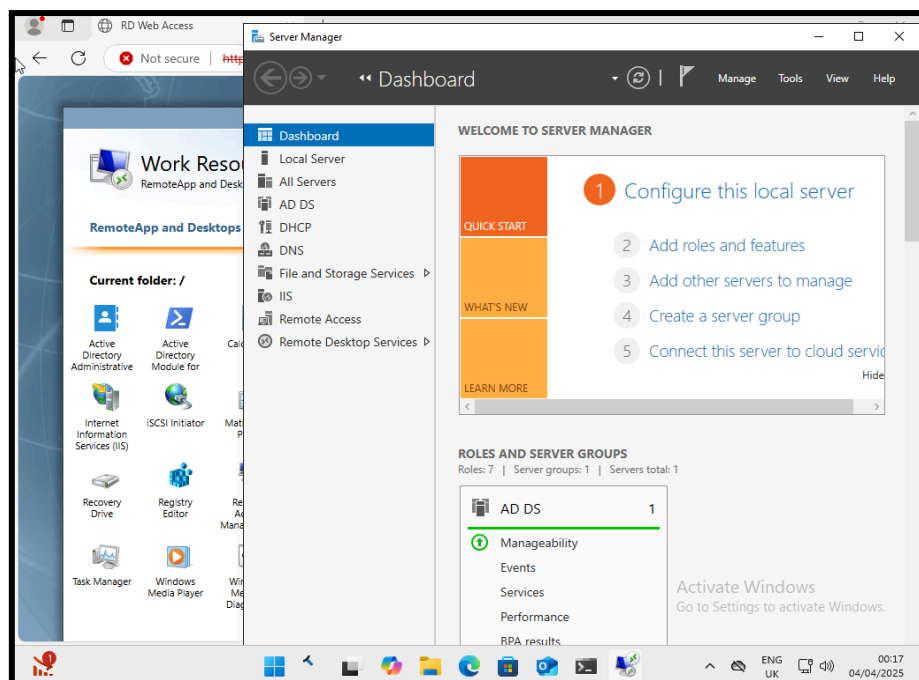
5. Testing the Access from Clients

The Rd Web interface was then tested from a client browser. After login, all published applications were displayed and could be launched.



[49] Successful connection to RD Web Access interface from a client browser

A user was also able to launch the Server Manager remotely through RDS, confirming that the applications were functional and responding.



[50] Server Manager opened remotely from the RDS portal

Conclusion

This project was a real challenge, but also a great opportunity to explore how different network services work together in a virtual environment. Starting from a basic setup, the system was extended step by step into a complete and realistic two-site infrastructure, capable of handling user management, secure communication, file sharing, web services, and remote access.

One of the most important outcomes is not just the final result, but the process itself. Understanding how systems interact, solving unexpected issues, and learning how to troubleshoot and adapt. Every part of the setup, from configuring trust between domains to making a remote application launch from a browser, required patience, precision, and logic.

In the end, the virtual network behaves like a real-world company network, and all critical services are in place. It proves that even with limited resources and a virtual lab, it's possible to simulate complex, professional-level infrastructure.