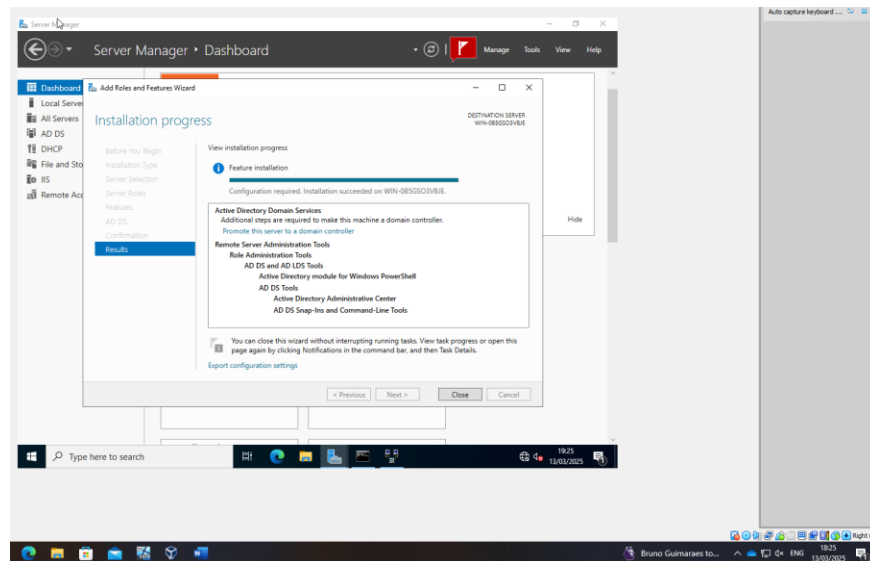


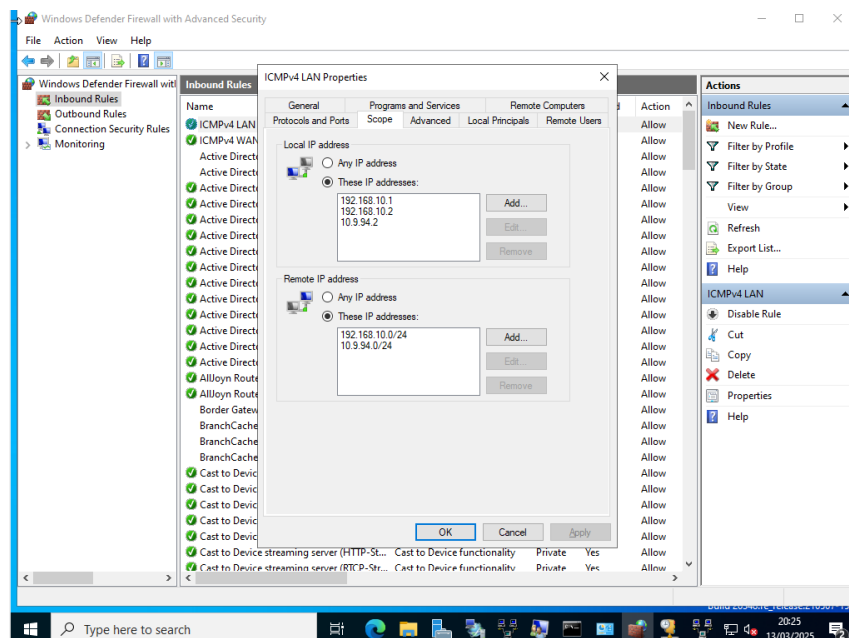
Network Operating Systems and Services – Coursework 2

(Thomas QIAN, 2422983)

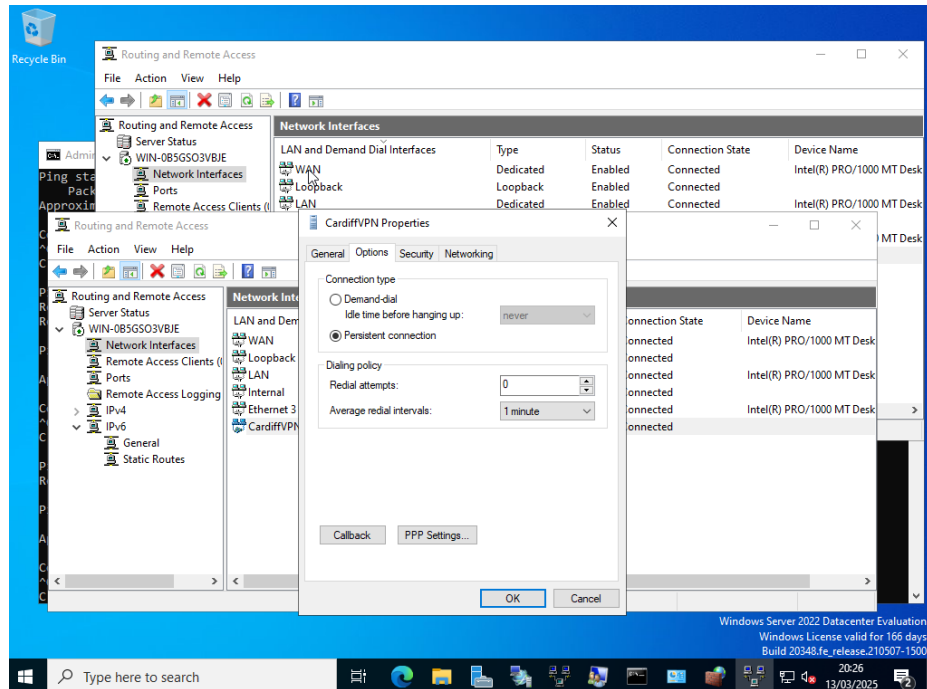
ADDS Installation



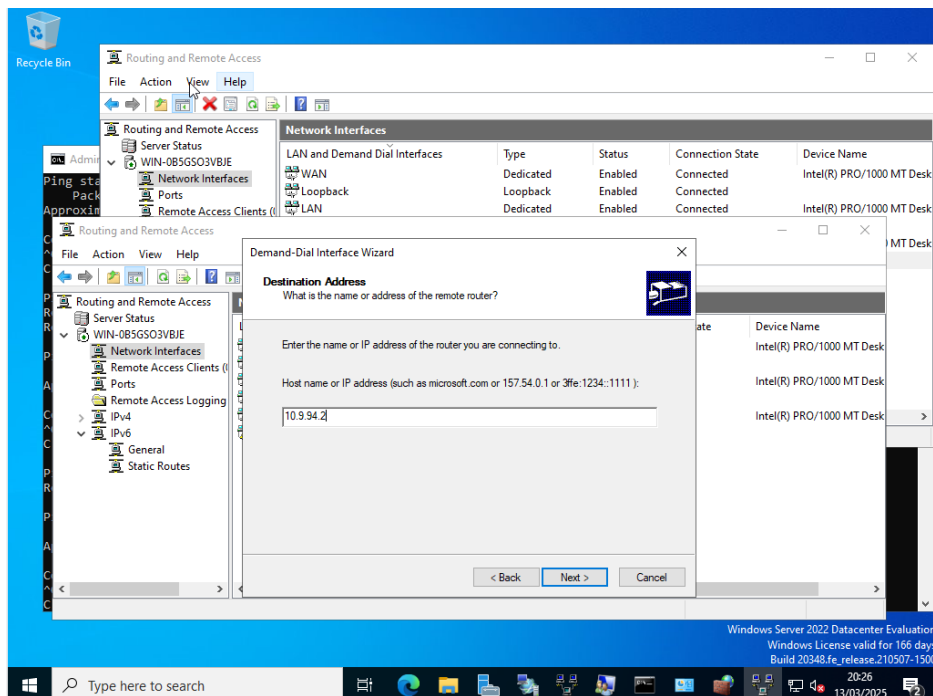
Allowing ping from remote of Cardiff LAN



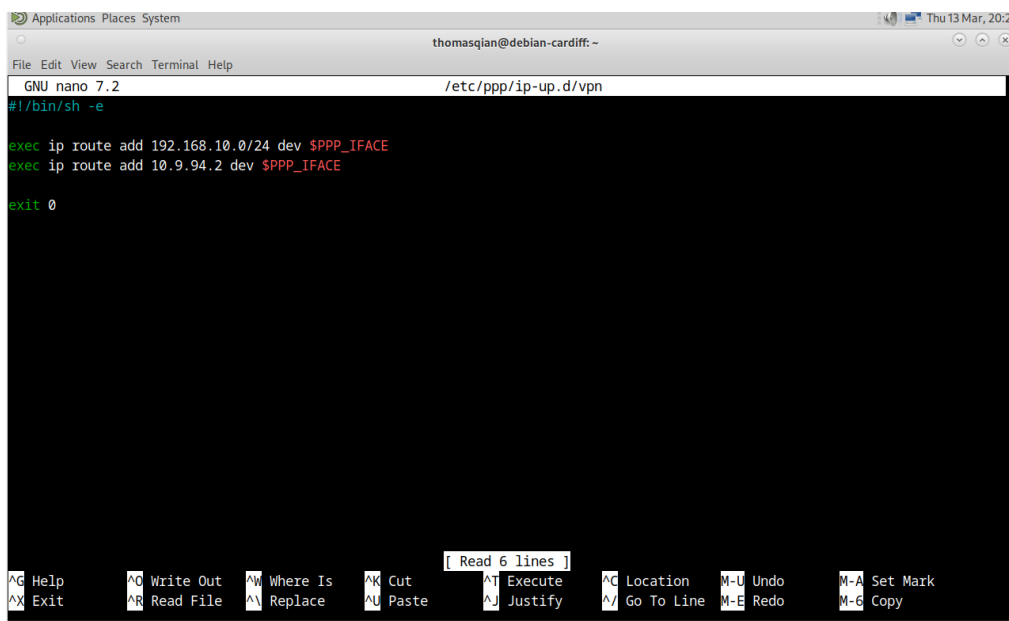
VPN creation Interface (Persistent Connection allowed)



Creating Dial in VPN Interface



Static Route for Swansea LAN in Debian



The screenshot shows a terminal window titled 'thomasqian@debian-cardiff: ~' with the file '/etc/ppp/ip-up.d/vpn' open in the GNU nano 7.2 editor. The editor's status bar indicates 'Read 6 lines'. The content of the file is as follows:

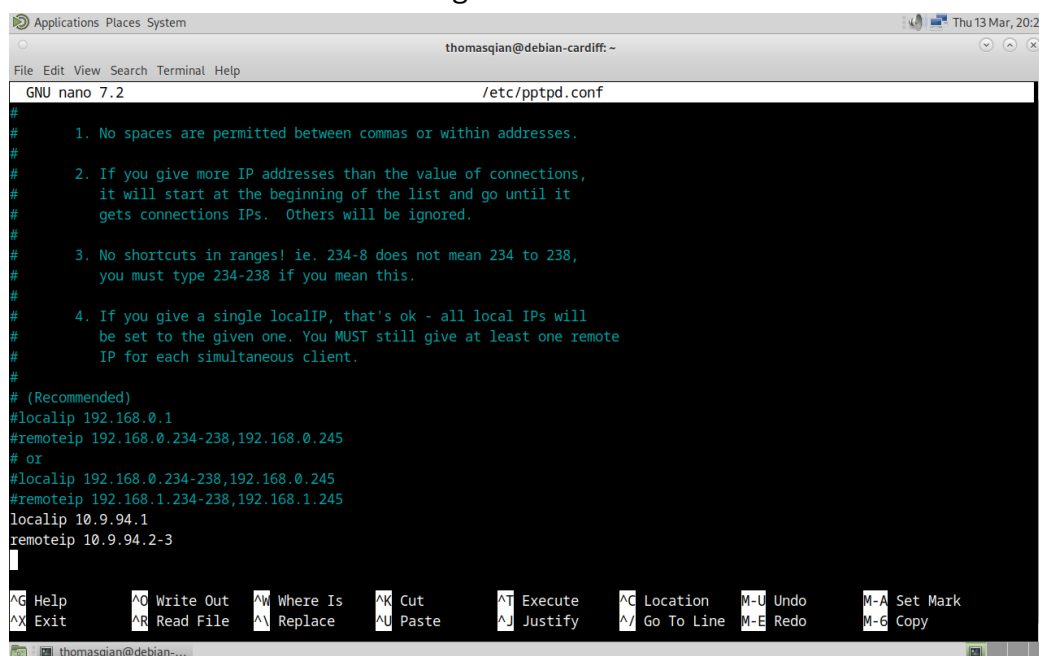
```
#!/bin/sh -e

exec ip route add 192.168.10.0/24 dev $PPP_IFACE
exec ip route add 10.9.94.2 dev $PPP_IFACE

exit 0
```

The bottom of the terminal shows a standard nano editor command palette with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Undo, Redo, Set Mark, and Copy.

PPTPD Configuration for Debian VPN

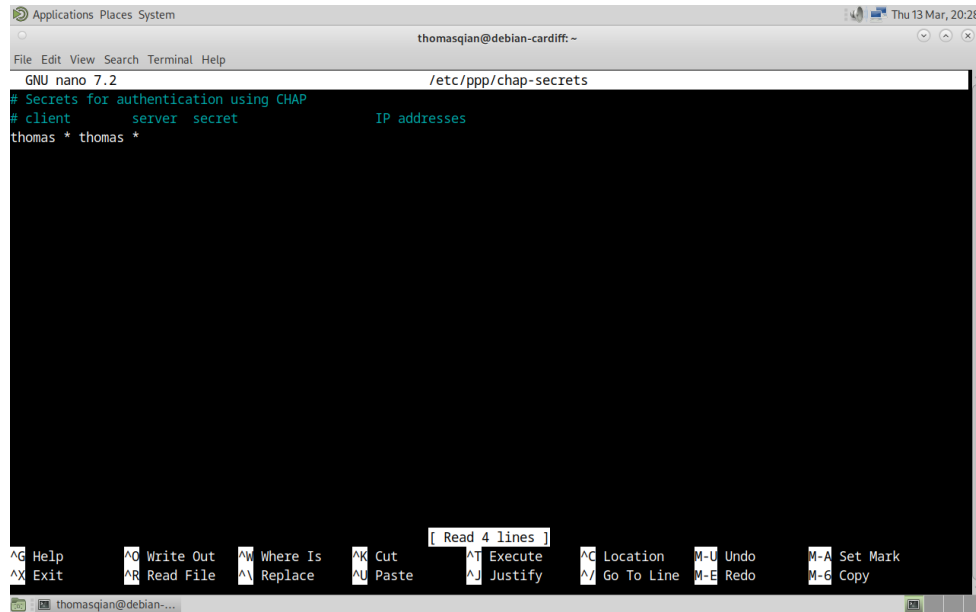


The screenshot shows a terminal window titled 'thomasqian@debian-cardiff: ~' with the file '/etc/pptpd.conf' open in the GNU nano 7.2 editor. The editor's status bar indicates 'Read 6 lines'. The content of the file is as follows:

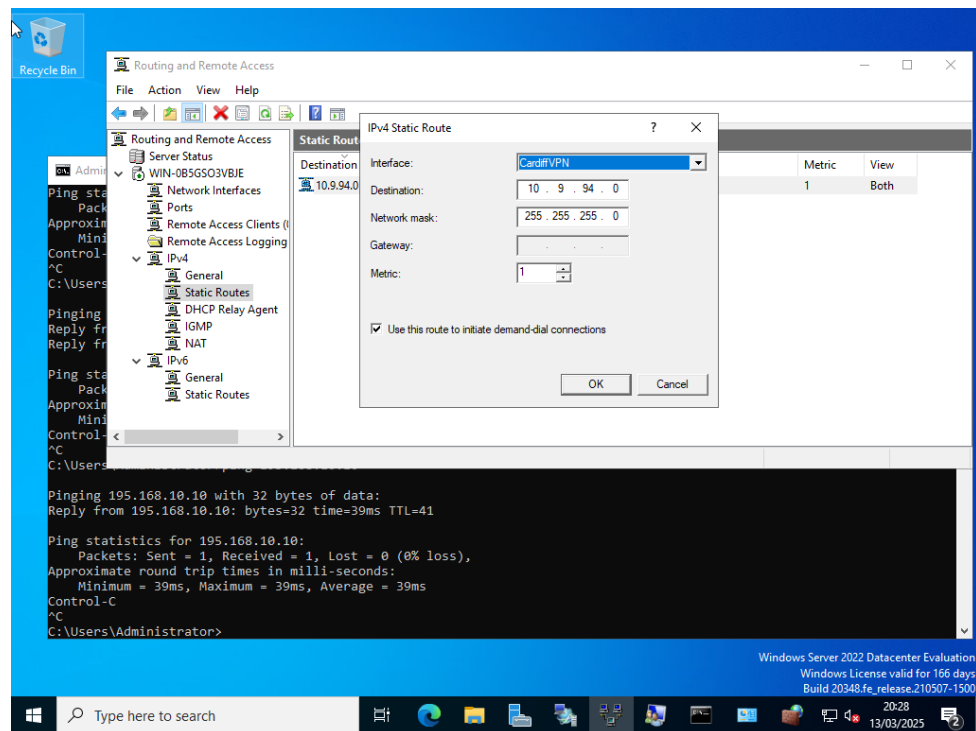
```
#
# 1. No spaces are permitted between commas or within addresses.
#
# 2. If you give more IP addresses than the value of connections,
#    it will start at the beginning of the list and go until it
#    gets connections IPs. Others will be ignored.
#
# 3. No shortcuts in ranges! ie. 234-8 does not mean 234 to 238,
#    you must type 234-238 if you mean this.
#
# 4. If you give a single localIP, that's ok - all local IPs will
#    be set to the given one. You MUST still give at least one remote
#    IP for each simultaneous client.
#
# (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 10.9.94.1
remoteip 10.9.94.2-3
```

The bottom of the terminal shows a standard nano editor command palette with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Undo, Redo, Set Mark, and Copy.

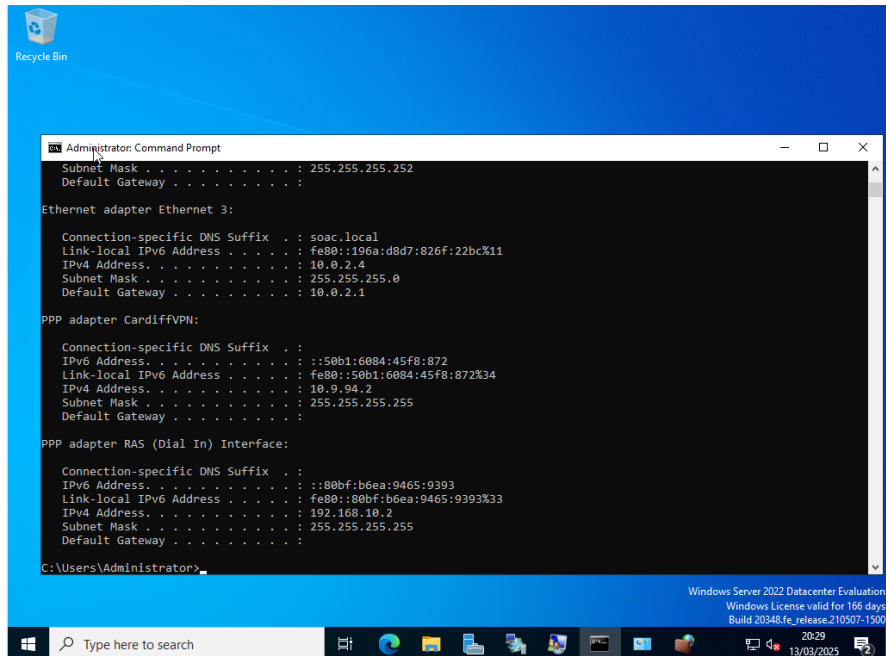
VPN User in Debian's PPP Interface



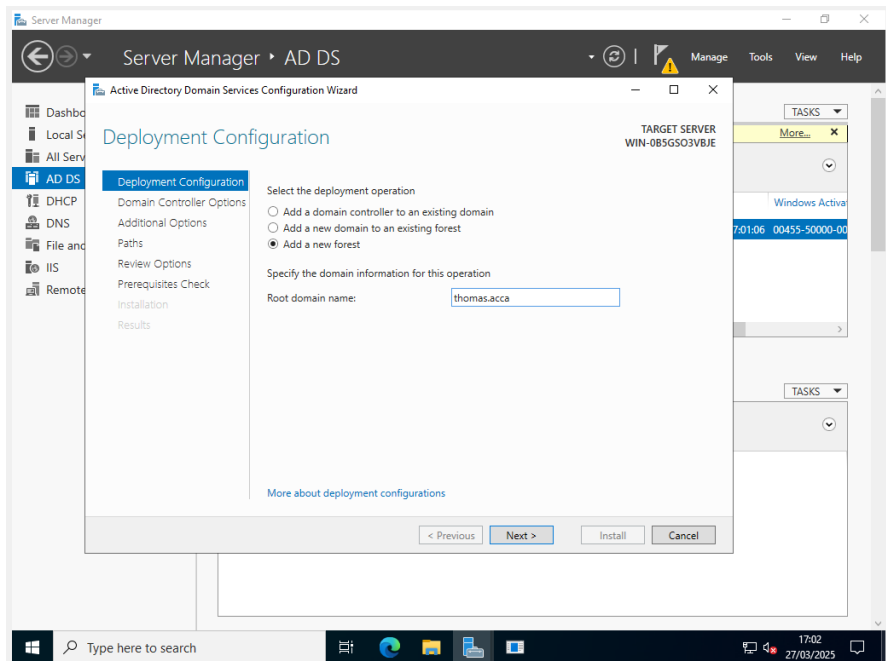
Static Route for CardiffVPN in Windows Server



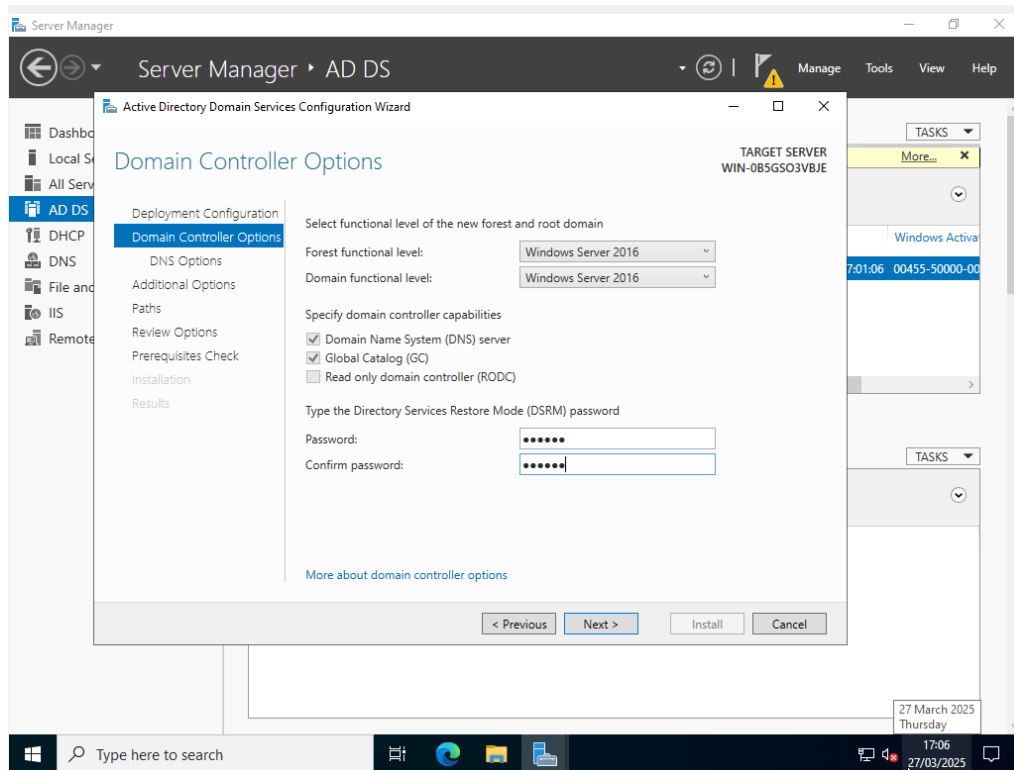
VPN Connected in Windows Server Successful



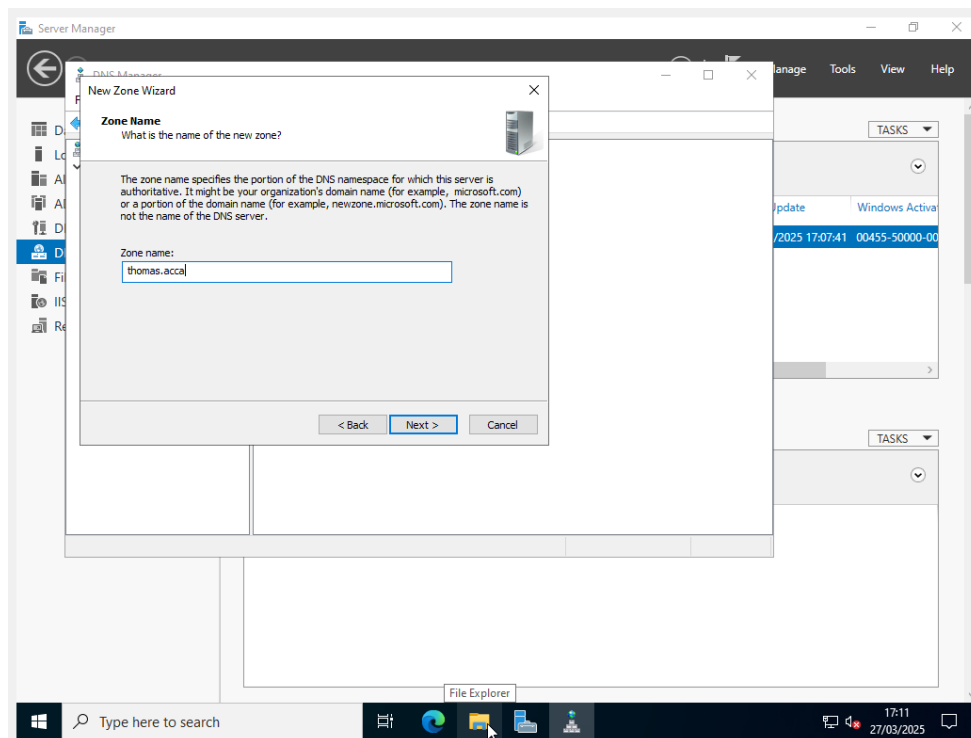
“thomasqian.acca” Forest creation



Creation of the password (Domain Controller Options)



Configuration of DNS Zone Name



Forest thomasqian.acca window and folders

The image consists of two screenshots of a Windows Server environment, specifically the Server Manager console.

Top Screenshot: The DNS Manager console is open, showing the hierarchy for the forest `thomasqian.acca`. The console tree on the left shows the following structure:

- DNS
- WIN-0B5GSO3VBIE.thomasqian.acca
- Forward Lookup Zones
- ._msdcs.thomasqian.acca
- thomasqian.acca
- ._msdcs
- ._sites
- ._tcp
- ._udp
- DomainDnsZones
- ForestDnsZones
- ._msdcs
- ._sites
- ._tcp
- Reverse Lookup Zones
- Trust Points
- Conditional Forwarders

The main pane displays a table of DNS records:

Name	Type	Data	Timestamp
._msdcs			
._sites			
._tcp			
._udp			
DomainDnsZones			
ForestDnsZones			
(same as parent folder)	Start of Authority (SOA)	[33] win-0b5gso3vbjie.tho...	static
(same as parent folder)	Name Server (NS)	win-0b5gso3vbjie.thomas...	static
(same as parent folder)	Host (A)	192.168.10.1	27/03/2025
win-0b5gso3vbjie	Host (A)	192.168.10.1	static

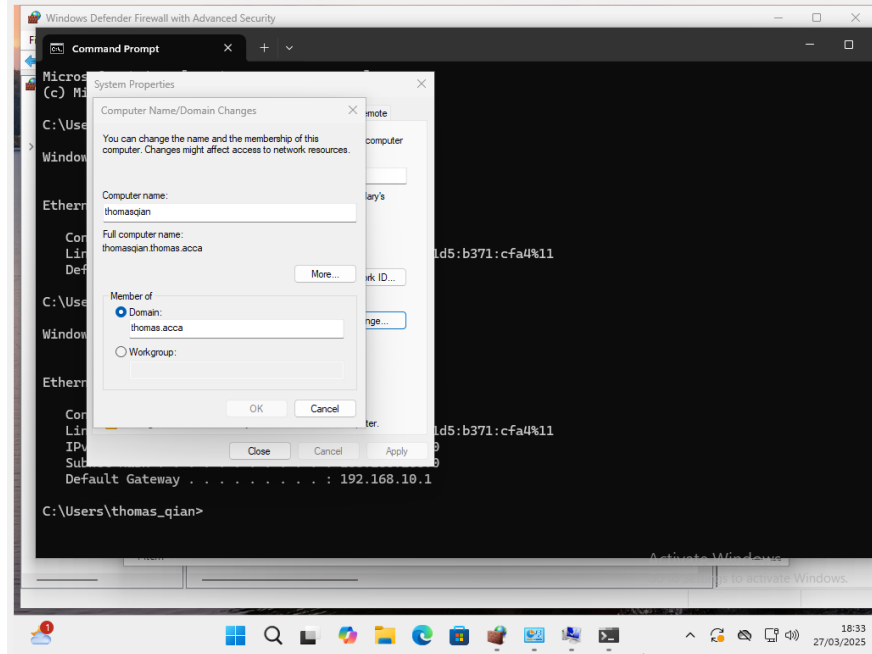
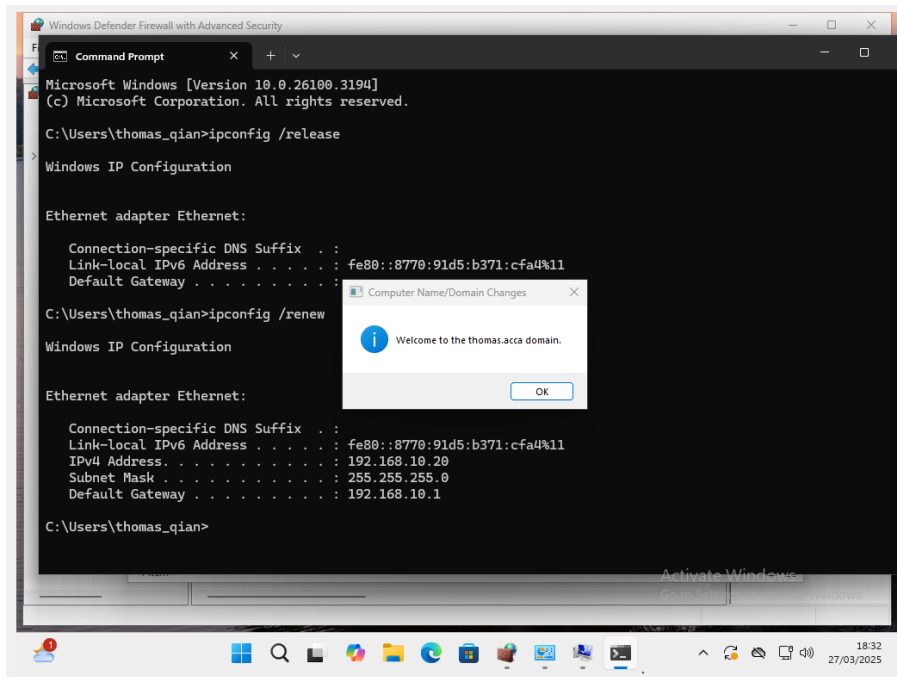
Bottom Screenshot: The "New Object - User" dialog box is open, showing the "Create in: thomasqian.acca/Users" path. The fields are filled with:

- First name: thomas
- Last name: qian
- Full name: thomas qian
- User login name: t.qian
- User login name (pre-Windows 2000): THOMAS\

The "Groups" list on the right shows the following groups:

- Group Polic... Security Group... Members in this group c...
- Guest User Built-in account for gue...
- Key Admins Security Group... Members of this group ...
- Protected Us... Security Group... Members of this group ...
- RAS and IAS ... Security Group... Servers in this group can...
- Read-only D... Security Group... Members of this group ...

Forest Domain working



Forest Viewed from Cardiff/Debian

```
Applications Places System
thomasqian@debian-cardiff: ~
File Edit View Search Terminal Help
GNU nano 7.2 /etc/resolv.conf *
nameserver 192.168.10.1
search thomas.acca
nameserver 10.9.94.1
search thomas.acca
nameserver 1.1.1.1
nameserver 8.8.8.8

Save modified buffer?
Y Yes
N No Cancel

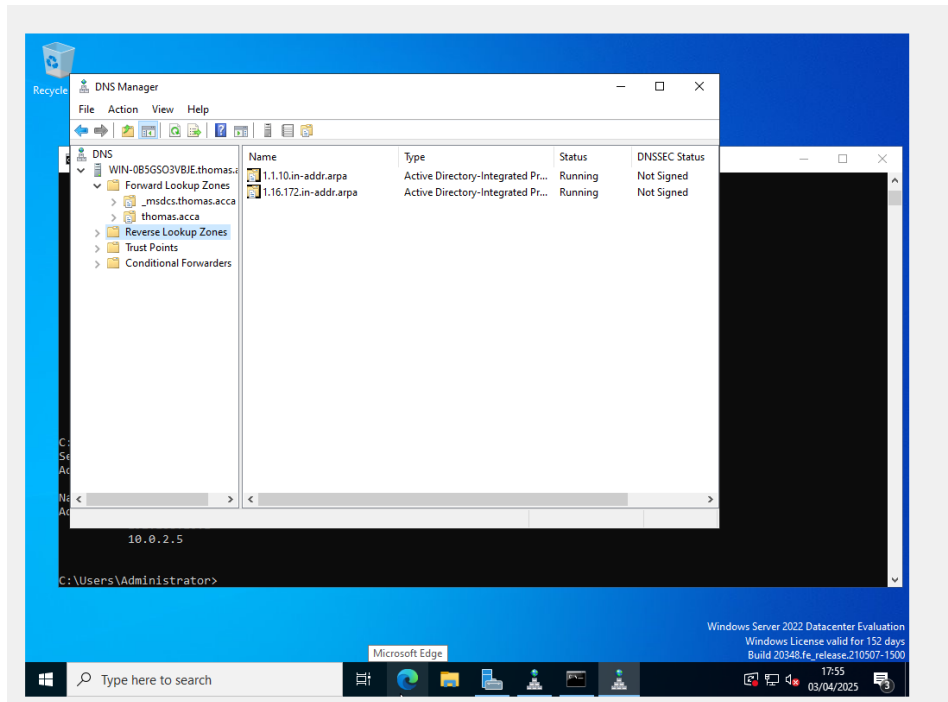
thomasqian@debian-cardiff:~$
thomasqian@debian-cardiff:~$ hostname
debian-cardiff
thomasqian@debian-cardiff:~$ hostname -f
debian-cardiff.soac.local
thomasqian@debian-cardiff:~$ domainname
(none)
thomasqian@debian-cardiff:~$
```

Host A and PTR Record

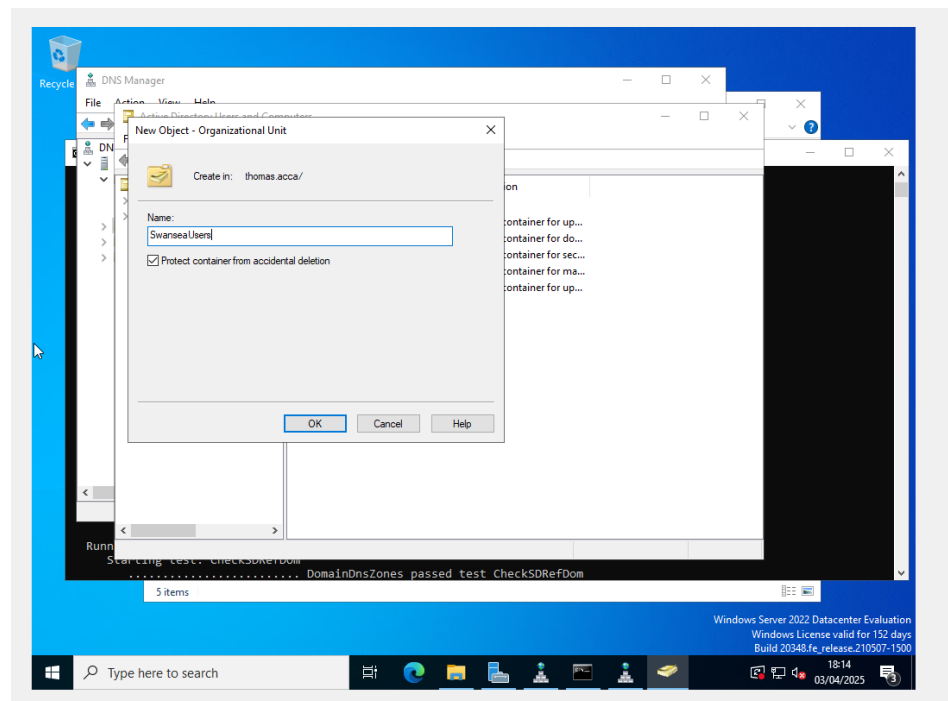
The screenshot shows the Windows DNS Manager interface. The left pane displays the tree structure of DNS zones, including Forward Lookup Zones, Reverse Lookup Zones, Trust Points, and Conditional Forwarders. The right pane shows the details of the Forward Lookup Zones, listing various Host (A) records and Pointer (PTR) records. The records are organized into a table with columns for Name, Type, Data, and Timestamp.

Name	Type	Data	Timestamp
(same as parent folder)	Start of Authority (SOA)	[1041] win-0b5gso3vbje...	static
(same as parent folder)	Name Server (NS)	win-0b5gso3vbje.thomas...	static
(same as parent folder)	Host (A)	192.168.10.1	27/03/202
(same as parent folder)	Host (A)	10.0.2.5	27/03/202
(same as parent folder)	Host (A)	210.94.10.1	27/03/202
(same as parent folder)	Host (A)	192.168.10.20	27/03/202
(same as parent folder)	Host (A)	210.94.10.1	static
(same as parent folder)	Host (A)	10.0.2.5	static
(same as parent folder)	Host (A)	192.168.10.1	static
(same as parent folder)	Host (A)	10.1.1.10	
(same as parent folder)	Host (A)	172.16.1.10	
(same as parent folder)	Pointer (PTR)	server10	
(same as parent folder)	Pointer (PTR)	server172	

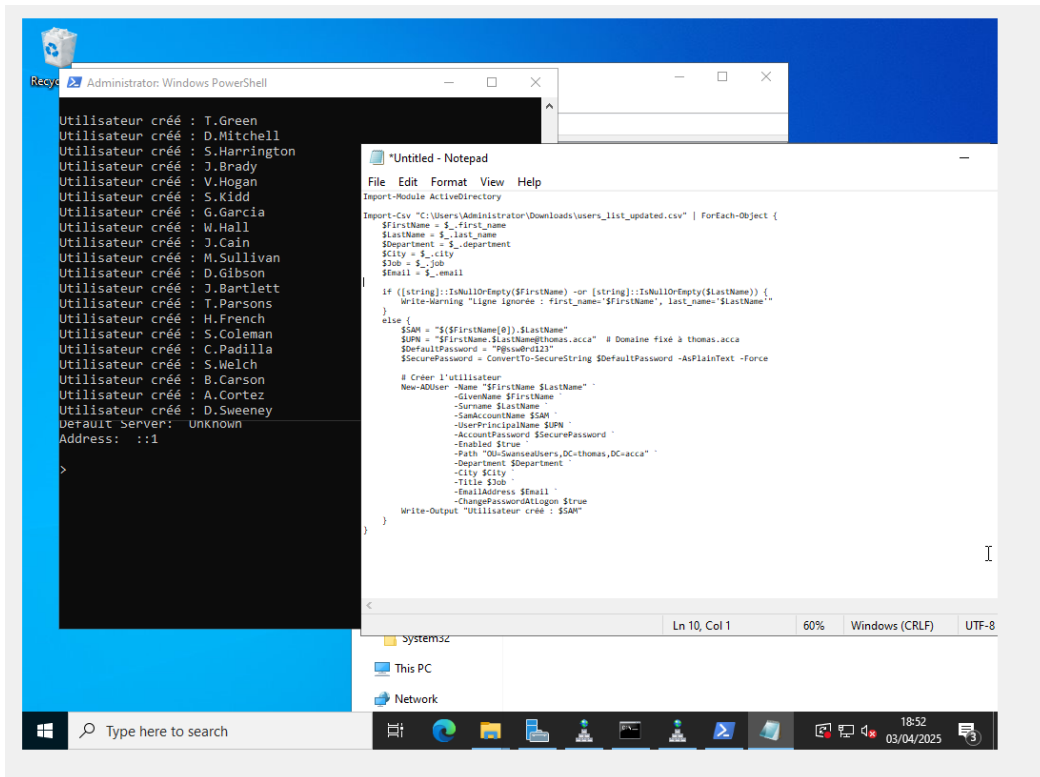
PTR Records (Reverse Lookup Zones)



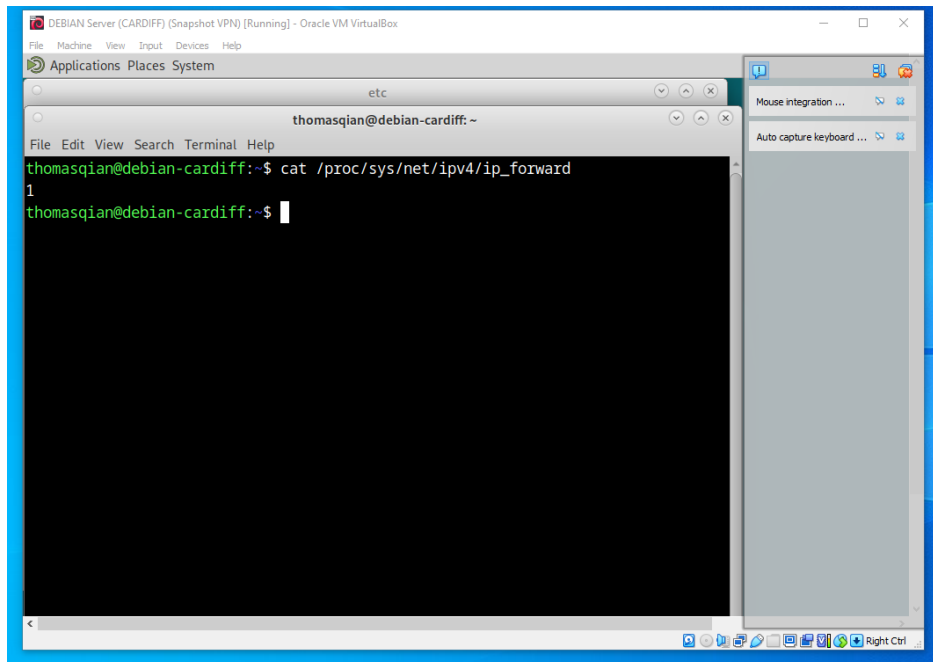
Organization Unit for Users



Creating Users and script with the help of generative AI



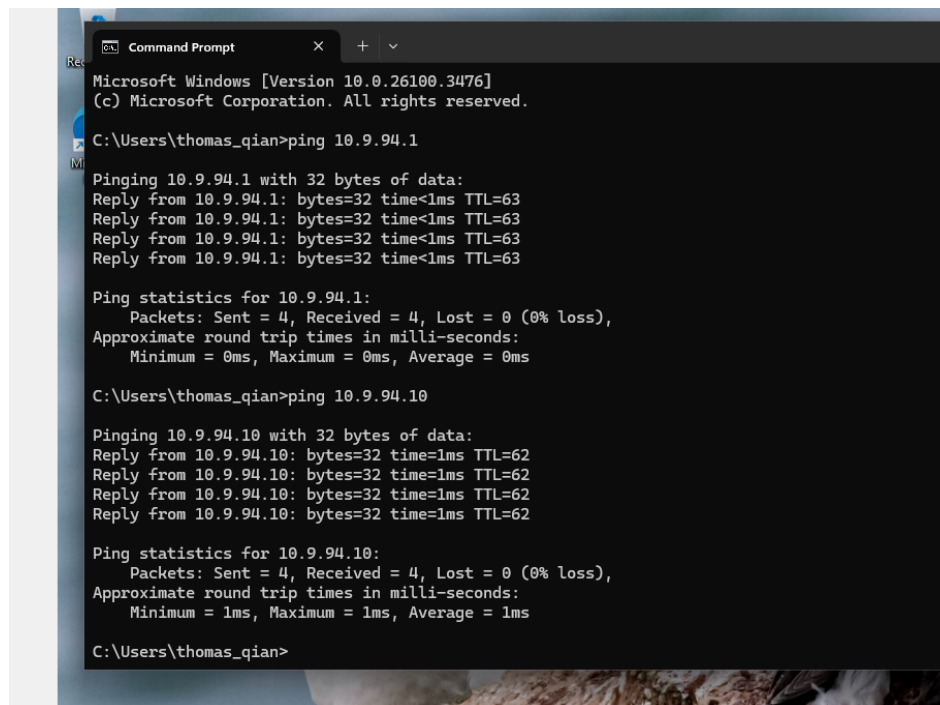
Routing test working



Windows Server and Client PING from Debian

```
thomasqian@debian-cardiff:~$ ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=1 ttl=127 time=0.501 ms
64 bytes from 192.168.10.1: icmp_seq=2 ttl=127 time=0.615 ms
64 bytes from 192.168.10.1: icmp_seq=3 ttl=127 time=0.414 ms
64 bytes from 192.168.10.1: icmp_seq=4 ttl=127 time=0.634 ms
^C
--- 192.168.10.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3057ms
rtt min/avg/max/mdev = 0.414/0.541/0.634/0.089 ms
thomasqian@debian-cardiff:~$ ping 192.168.10.20
PING 192.168.10.20 (192.168.10.20) 56(84) bytes of data.
64 bytes from 192.168.10.20: icmp_seq=1 ttl=127 time=0.773 ms
64 bytes from 192.168.10.20: icmp_seq=2 ttl=127 time=0.785 ms
64 bytes from 192.168.10.20: icmp_seq=3 ttl=127 time=0.627 ms
64 bytes from 192.168.10.20: icmp_seq=4 ttl=127 time=0.572 ms
^C
--- 192.168.10.20 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3036ms
rtt min/avg/max/mdev = 0.572/0.689/0.785/0.091 ms
```

Debian Server and Ubuntu Client PING from Windows Server (Connectivity complete)



```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\thomas_qian>ping 10.9.94.1

Pinging 10.9.94.1 with 32 bytes of data:
Reply from 10.9.94.1: bytes=32 time<1ms TTL=63
Reply from 10.9.94.1: bytes=32 time<1ms TTL=63
Reply from 10.9.94.1: bytes=32 time<1ms TTL=63
Reply from 10.9.94.1: bytes=32 time<1ms TTL=63

Ping statistics for 10.9.94.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

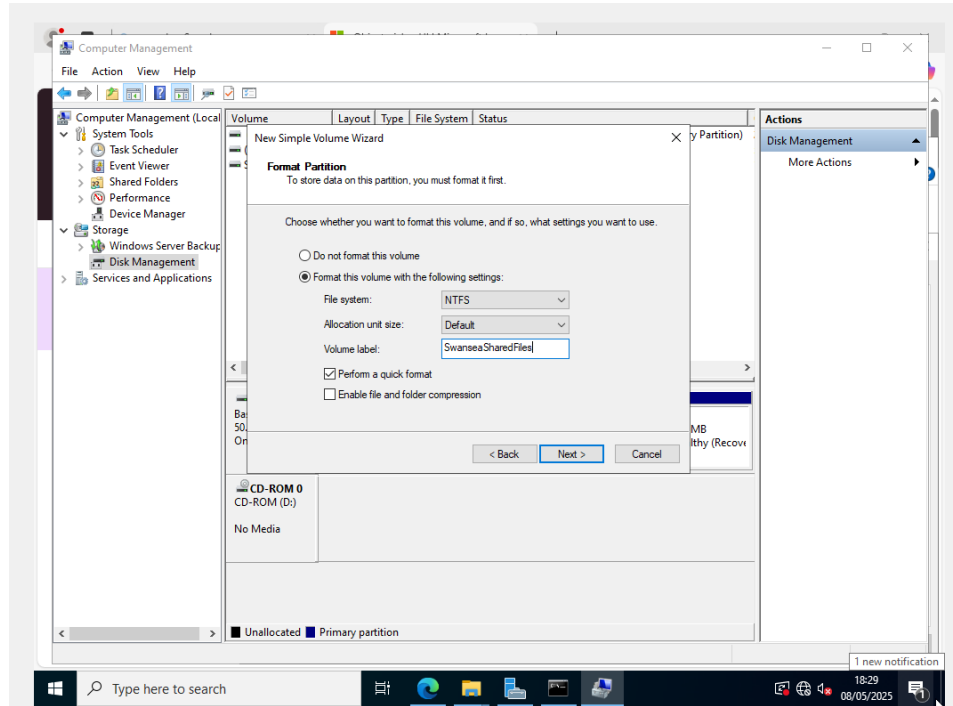
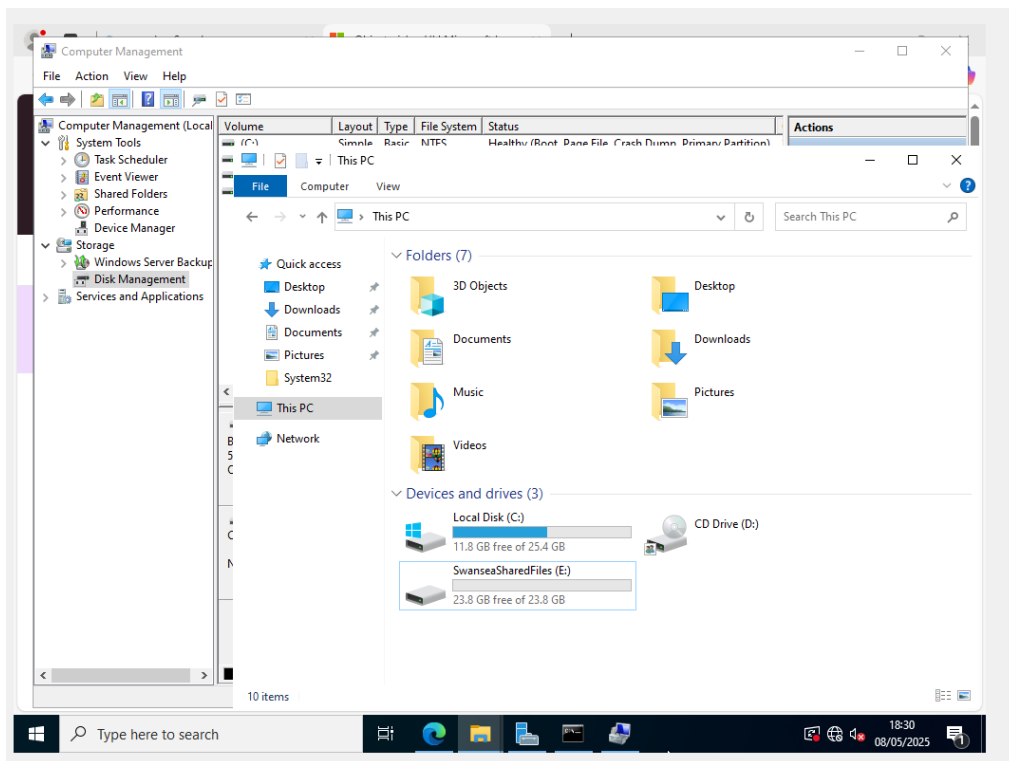
C:\Users\thomas_qian>ping 10.9.94.10

Pinging 10.9.94.10 with 32 bytes of data:
Reply from 10.9.94.10: bytes=32 time=1ms TTL=62
Reply from 10.9.94.10: bytes=32 time=1ms TTL=62
Reply from 10.9.94.10: bytes=32 time=1ms TTL=62
Reply from 10.9.94.10: bytes=32 time=1ms TTL=62

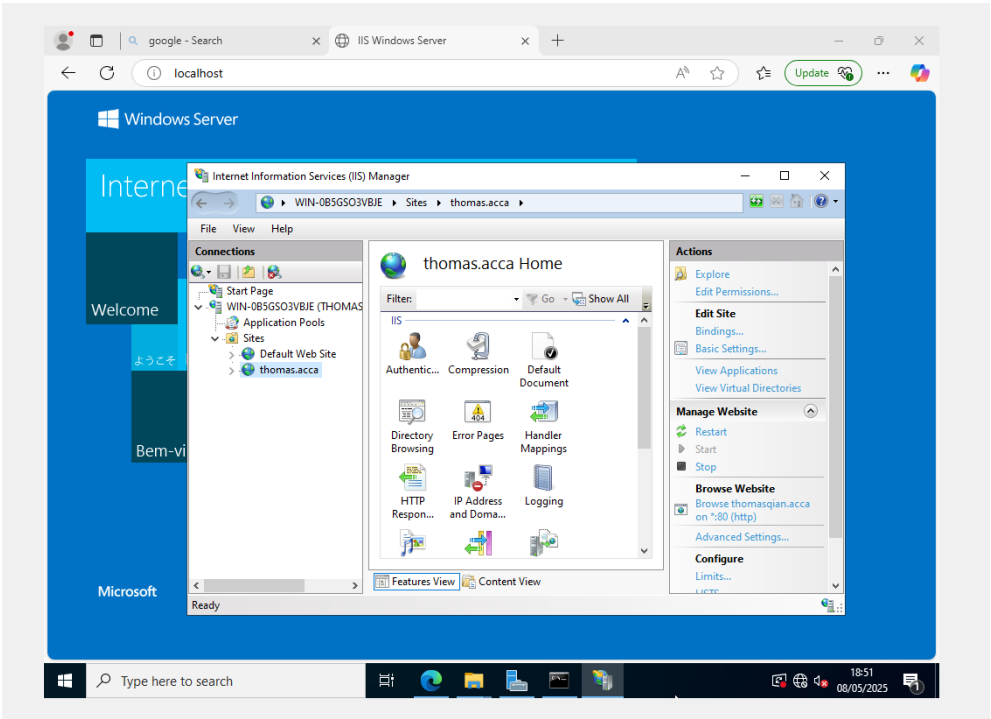
Ping statistics for 10.9.94.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 1ms, Average = 1ms

C:\Users\thomas_qian>
```

Shared Files Creation Disk



thomasqian.acca website creation



SCRIPTS

SCRIPT 1:

```
Script_1.ps1 X Script_2.ps1 Script_3.ps1

1 # PowerShell script to check
2
3 #In Practical Application
4 #1. RRAS, ICMP Firewall rules, (5 Marks)
5 #2 and 3. domain controller, and domain members (5 + 5 Marks)
6
7 # Run on S-DC (Suansea DC) with administrative privileges
8 # Verifies configurations without making changes
9
10 # Set error handling
11 $ErrorActionPreference = "Stop"
12
13 # Print execution date and time
14 $executionTime = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
15 Write-Log "Script executed on: $executionTime"
16 Write-Log "Starting assessment of RRAS, domain controller, and domain members..."
17
18 # Log file setup
19 $logFile = "C:\CheckRRASDomainandMembers_Log.txt"
20 Function Write-Log {
21     param($message)
22     $timestamp = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
23     "$timestamp - $message" | Out-File -FilePath $logFile -append
24     Write-Host $message
25 }
26
27 Write-Log "Starting assessment of RRAS, domain controller, and domain members..."
28
29 # Define domain parameters
30 $domainName = "thomas.acca"
31
32 # 1. Check Routing and Remote Access Service (RRAS) Installation
33 Write-Log "Checking Routing and Remote Access Service (RRAS)..."
34 $rras = Get-Service -Name RemoteAccess -ErrorAction SilentlyContinue
35 If ($rras -and $rras.Status -eq "Running") {
36     Write-Log "RRAS is installed and running."
37 } Else {
38     Write-Log "ERROR: RRAS is not installed or not running."
39 }
40
41 # Check ICMP Firewall Rules
42 Write-Log "Checking ICMP Firewall rules..."
43 $icmpRule = Get-NetFirewallRule | Where-Object {
44     $_.DisplayName -like "ICMP*" -and
45     $_.Enabled -eq $true -and
46     $_.Protocol -eq "ICMPv4" -and
47     $_.Action -eq "Allow"
48 }
49 If ($icmpRule) {
50     Write-Log "ICMPv4 firewall rule is enabled and allows traffic."
51 } Else {
52     Write-Log "ERROR: ICMPv4 firewall rule is not enabled or not configured to allow traffic."
53 }
54
55 # 2. Check Domain Controller for thomas.acca
56 Write-Log "Checking domain controller for $domainName..."
57 $forest = Get-ADForest -ErrorAction SilentlyContinue
58 $domain = Get-ADDomain -ErrorAction SilentlyContinue
59 If ($forest -and $forest.Name -eq $domainName -and $domain -and $domain.DNSRoot -eq $domainName) {
60     $dc = Get-ADDomainController -Filter { HostName -like "WIN-06SG03V63E*" } -ErrorAction SilentlyContinue
61     If ($dc) {
62         Write-Log "Domain controller for $domainName is operational on S-DC."
63     } Else {
64         Write-Log "ERROR: Domain controller for $domainName not found on S-DC."
65     }
66 } Else {
67     Write-Log "ERROR: Forest or domain $domainName not found."
68 }
69
70 # 3. List Computers and Users joined to the Domain
71 Write-Log "Listing computers and users joined to $domainName..."
72
73 # Get computers
74 $computers = Get-ADComputer -Filter "Properties" -ErrorAction SilentlyContinue
75 $computerCount = ($computers | Measure-Object) Count
76 Write-Log "Number of computers joined to $domainName: $computerCount"
77 If ($computerCount -gt 0) {
78     Write-Log "Computers joined to the domain:"
79     foreach ($computer in $computers) {
80         Write-Log " - $($computer.Name) (DNS: $($computer.DNSHostName))"
81     }
82 } Else {
83     Write-Log "WARNING: No computers joined to $domainName."
84 }
85
86 # Get users (excluding default accounts like Administrator)
87 $users = Get-ADUser -Filter { Enabled -eq $true } -Properties * |
88 Where-Object { $_.SamAccountName -notlike "Administrator" -and $_.SamAccountName -notlike "krbtgt" }
89 $userCount = ($users | Measure-Object) Count
90 Write-Log "Number of user accounts in $domainName: $userCount"
91 If ($userCount -gt 0) {
92     Write-Log "User accounts in the domain:"
93     foreach ($user in $users) {
94         Write-Log " - $($user.SamAccountName) (Name: $($user.Name))"
95     }
96 } Else {
97     Write-Log "WARNING: No user accounts found in $domainName."
98 }
99
100 Write-Log "Assessment of RRAS, domain controller, and domain members completed. Review $logFile for details."
```

SCRIPT 2

```

Administrator: Windows PowerShell ISE

File Edit View Tools Debug Add-ons Help

Get-NetIPAddress -All -Detailed

Script: ps1 Script: ps1 X Script: ps1

# PowerShell script to list DNS records, reverse lookup records, and DHCP pools for specified IP ranges

# DNS practical application
# DNS records - appropriate records and reverse lookup zones. (5 marks)
# # and 20 - DHCP (5 - 3 marks)

# Run on 3-00 (change DNS with administrative privileges)
# List records only for specified IP ranges, ignoring others
# Records available only without displaying errors
# Print execution date and time in script

# Configure IP address ranges (modify, then as needed)
$IPranges = "0", "192.168.1.0/24", "10.0.0.0/24", "172.16.0.0/24"
# $IPranges = "0", "192.168.1.0/24", "192.168.1.0/24", "192.168.1.0/24"

# Set error handling to continue
Set-ErrorActionPreference -Continue

# Log file setup
$logfile = "C:\CheckDNSandDHCPLog.txt"
function Log {
    param($message)
    Write-Host -ForegroundColor Green "Log file: $message"
    Write-Host -ForegroundColor Green "Log file: $message"
    Write-Host -ForegroundColor Green "Log file: $message"
}

# Print execution date and time
Write-Host -ForegroundColor Green "Log file: $message"
Write-Host -ForegroundColor Green "Log file: $message"
Write-Host -ForegroundColor Green "Log file: $message"

# Define domain and reverse lookup zone parameters
$DomainName = "192.168.1.0/24"
$ReverseZoneName = "192.168.1.0/24"

# 1. Check DNS Service
Write-Host "Checking DNS service status..."
$Service = Get-Service -Name "DNS"
if ($Service.Status -ne "Running") {
    Write-Host "DNS service is not running."
} else {
    Write-Host "DNS service is available."
}

# 2. List DNS records for specified IP ranges
Write-Host "Listing DNS records for IP ranges: $($IPranges -join ', ')"
$Records = Get-DnsRecordSet -ZoneName $DomainName -ErrorAction SilentlyContinue
if ($Records) {
    foreach ($Record in $Records) {
        Write-Host "Record: $($Record.Name) - $($Record.Type) - $($Record.Value)"
    }
} else {
    Write-Host "No records found for specified IP ranges."
}

# 3. Create reverse lookup zones for specified IP ranges
Write-Host "Creating reverse lookup zones for IP ranges: $($IPranges -join ', ')"
$ReverseZones = Get-DnsRecordSet -ZoneName $DomainName -ErrorAction SilentlyContinue
if ($ReverseZones) {
    foreach ($Zone in $ReverseZones) {
        Write-Host "Zone: $($Zone.Name) - $($Zone.Type) - $($Zone.Value)"
    }
} else {
    Write-Host "No reverse lookup zones found for specified IP ranges."
}

# 4. Add reverse lookup records for specified IP ranges
Write-Host "Adding reverse lookup records for IP ranges: $($IPranges -join ', ')"
$ReverseRecords = Get-DnsRecordSet -ZoneName $DomainName -ErrorAction SilentlyContinue
if ($ReverseRecords) {
    foreach ($Record in $ReverseRecords) {
        Write-Host "Record: $($Record.Name) - $($Record.Type) - $($Record.Value)"
    }
} else {
    Write-Host "No reverse lookup records found for specified IP ranges."
}

```

The screenshot shows a Windows PowerShell console window with the title bar "Administrator: Windows PowerShell ISE". The menu bar includes "File", "Edit", "View", "Tools", "Debug", "Add-ons", and "Help". The toolbar contains icons for file operations and execution. The script is divided into three sections: "Script_1.ps1", "Script_2.ps1", and "Script_3.ps1".

The script content is as follows:

```

67
68 # 3. List Reverse Lookup (PTR) Records for Specified IP Ranges
69 3. List Reverse Lookup (PTR) records for specified IP ranges...
70 Foreach ($zone in $serversZones) {
71     Write-Log "Checking Reverse Lookup zone: $zone"
72     $serversDomainName = Get-SubserverZone -ZoneName $zone
73     If ($serversDomainName) {
74         $ptrRecords = Get-SubserverResourceRecord -ZoneName $zone -KeyType Ptr
75         If ($ptrRecords) {
76             Write-Log "PTR records in $zone: Available $($ptrRecords.Count) records"
77             Foreach ($record in $ptrRecords) {
78                 Write-Log " - HostName: $($record.HostName)"
79                 Write-Log " - PTR Domain Name: $($record.PtrDomainName)"
80                 Write-Log " - Timestamp: $($record.Timestamp)"
81                 Write-Log " - TTL: $($record.TTL)"
82             }
83         }
84     } else {
85         Write-Log "PTR records in $zone: Not available"
86     }
87 }
88 Write-Log "Reverse lookup zone $zone: not available"
89
90
91 # 4. Check DHCP Service
92 Write-Log "Checking DHCP service status..."
93 $dhcpService = Get-Service -Name "DHCPService"
94 If ($dhcpService -and $dhcpService.Status -ne "Running") {
95     Write-Log "DHCP service: Available"
96 } else {
97     Write-Log "DHCP service: Not available"
98 }
99
100
101 # 5. List DHCP Scopes for Specified IP Ranges
102 Write-Log "Listing DHCP scopes for IP ranges: $($IPRanges -join ", ")."
103 $dhcpScopes = Get-DhcpServerV4Scope
104 Where-Object { $_.ScopeAddress -match "^(192.168.1.1|192.168.1.1|192.0.0.0|0.0.0.0)" }
105 If ($dhcpScopes) {
106     Write-Log "DHCP scopes for specified IP ranges: Available $($dhcpScopes.Count) scopes"
107     Foreach ($scope in $dhcpScopes) {
108         Write-Log " - DHCP Scope Details:"
109         Write-Log " - Scope ID: $($scope.ScopeID)"
110         Write-Log " - Name: $($scope.Name)"
111         Write-Log " - Start Range: $($scope.StartRange)"
112         Write-Log " - End Range: $($scope.EndRange)"
113         Write-Log " - Subnet Mask: $($scope.SubnetMask)"
114         Write-Log " - State: $($scope.State)"
115         Write-Log " - Description: $($scope.Description)"
116         Write-Log " - Options: $($scope.Options | Get-ObjectProperty -ScopeId $scope.ScopeId | Get-ObjectProperty -OptionName $scope.Options)"
117         Write-Log " - DHCP Options:"
118         Foreach ($option in $scope.Options) {
119             Write-Log " - Option ID: $($option.OptionID) ($($option.Name))"
120             Write-Log " - Value: $($option.Value) ($($option.Value -join ", "))"
121         }
122     }
123 } else {
124     Write-Log " - No DHCP options configured."
125 }
126
127
128 } else {
129     Write-Log "DHCP scopes for specified IP range: Not available"
130 }
131
132 Write-Log "Assessment of DNS records and DHCP pools completed. Review $logfile for details."

```

The console output shows the execution of the script, including the assessment of DNS records and DHCP pools. The final output is "Assessment of DNS records and DHCP pools completed. Review \$logfile for details."

At the bottom of the window, there are two buttons: "Completed" and "Task View".

SCRIPT 3

```
Administrator: Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help

Script_1.ps1 Script_2.ps1 Script_3.ps1 X

1 # PowerShell script to test network connectivity, disk drive availability, IIS authentication, and RDS configuration
2
3 # In Practical Application
4 # IIS Test Connection (5 Marks)
5 # In Knowledge and Understanding, CIS
6 # 1, 2, 3 - Sharing Disk Drive, IIS Server, RDS (5 + 5 Marks)
7 # Execute this script and then manually check and assign marks as per the attempted configuration.
8 # Do necessary changes like ip address, names accordingly before execution
9
10
11 # Run on S-DC (Swansea DC) with administrative privileges
12 # Tests ping between server and client, tracer via VPN, disk share, IIS, and RDS
13 # Reports availability without displaying errors
14 # Prints execution date and time in output
15
16 # Set error handling to silent
17 $ErrorActionPreference = "SilentlyContinue"
18
19 # Log File setup
20 $LogFile = "C:\CheckNetworkConnectivity_Log.txt"
21
22 -Function Write-Log {
23     param($Message)
24     $TimeStamp = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
25     "Timestamp = $Message" | Out-File -FilePath $LogFile -Append
26     Write-Host $TimeStamp - $Message
27 }
28
29 # Print execution date and time
30 $ExecutionTime = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
31 Write-Log "Script executed on: $ExecutionTime"
32 Write-Log "Starting assessment of network connectivity, disk drive, IIS, and RDS configurations..."
33
34 # Define network parameters
35 $DomainName = "thomas.acca"
36 $SwanseaServer = "WIN-DESGO3V83S" # 192.168.1.1
37 $SwanseaClient = "thomasqlan" # 192.168.1.22 (DHCP)
38 $CardiffServer = "Debian-Cardiff" # 192.168.2.20 (Debian Server)
39 $CardiffClient = "thomasqlan-cardiff" # 192.168.2.20 (DHCP)
40 $VpnGateway = "10.0.2.1" # Cardiff VPN gateway, assuming Site-to-Site VPN
41
42 # 1. Test Ping Connectivity (Swansea: S-DC to ClientS1, Cardiff: C-S1 to ClientC1)
43 Write-Log "Testing ping connectivity in Swansea network ($SwanseaServer to $SwanseaClient)..."
44 $SwanseaPing = Test-Connection -ComputerName $SwanseaClient -Count 4 -Quiet
45 if ($SwanseaPing) {
46     Write-Log "Ping from $SwanseaServer to $SwanseaClient: Available"
47 } else {
48     Write-Log "Ping from $SwanseaServer to $SwanseaClient: Not Available"
49 }
50
51 Write-Log "Testing ping connectivity in Cardiff network ($CardiffServer to $CardiffClient)..."
52 $CardiffPing = Test-Connection -ComputerName $CardiffClient -Count 4 -Quiet
53 if ($CardiffPing) {
54     Write-Log "Ping from $CardiffServer to $CardiffClient: Available"
55 } else {
56     Write-Log "Ping from $CardiffServer to $CardiffClient: Not Available"
57 }
58
59 # 2. Perform Traceroute from S-DC to Debian-Server (C-S1) to check VPN Usage
60 Write-Log "Performing traceroute from $SwanseaServer to $CardiffServer to verify VPN usage..."
61 $TracerouteResult = Test-NetConnection -ComputerName $CardiffServer -TraceRoute
62 if ($TracerouteResult.TraceRoute) {
63     Write-Log "Traceroute to $CardiffServer: Available"
64     Write-Log "Traceroute hops:"
65     foreach ($hop in $TracerouteResult.TraceRoute) {
66         Write-Log "$hop"
67     }
68 } else {
69     Write-Log "Traceroute to $CardiffServer: Not Available"
70 }
71
72 # 3. Check Disk Drive (D:\SwanseaFiles) Availability
73 Write-Log "Checking D:\SwanseaFiles share availability for Swansea and Cardiff networks..."
74 $Share = Get-WebShare -Name "SwanseaFiles"
75 if ($Share) {
76     $ShareAccess = Get-SmbShareAccess -Name "SwanseaFiles"
77     $DomainUsersAccess = $ShareAccess | where-Object { $_.AccountName -like "$($DomainName)\Domain Users*" -and $_.AccessRight -eq "Read" }
78     if ($DomainUsersAccess) {
79         Write-Log "D:\SwanseaFiles share: Available (Read access for Domain Users)"
80     } else {
81         Write-Log "D:\SwanseaFiles share: Not Available (No read access for Domain Users)"
82     }
83 } else {
84     Write-Log "D:\SwanseaFiles share: Not Available"
85 }
86
87 # 4. Check IIS Server with Internal authentication
88 Write-Log "Checking IIS server with internal authentication for https://webs.$($DomainName)..."
89 $IISService = Get-Service -Name "IIS"
90 if ($IISService -and $IISService.Status -eq "Running") {
91     Write-Log "IIS service: Available"
92     $WebSite = Get-WebSite -Name "webs.$($DomainName)"
93     if ($WebSite) {
94         $Auth = Get-WebConfigurationProperty -Filter "/system.webServer/security/authentication/windowsAuthentication" -Name enabled -PSPath "IIS::Sites\webs.$($DomainName)"
95         if ($Auth) {
96             Write-Log "IIS website https://webs.$($DomainName) with Windows authentication: Available"
97             try {
98                 $Response = Invoke-WebRequest -Uri "https://webs.$($DomainName)" -UseDefaultCredentials
99                 if ($Response.StatusCode -like "2*" -and $Response.Headers["Server"] -like "Microsoft-IIS/10.0") {
100                     Write-Log "IIS website content: 'Welcome to Swansea Website': Available"
101                 } else {
102                     Write-Log "IIS website content: 'Welcome to Swansea Website': Not Available"
103                 }
104             } catch {
105                 Write-Log "Website access https://webs.$($DomainName): Not Available"
106             }
107         } else {
108             Write-Log "IIS website https://webs.$($DomainName) with Windows authentication: Not Available"
109         }
110     } else {
111         Write-Log "IIS website webs.$($DomainName): Not Available"
112     }
113 } else {
114     Write-Log "IIS service: Not Available"
115 }
116
117 # 5. Check S-DC Remote Access via https://rds.id.nithesh.acca and Certificates
118 Write-Log "Checking S-DC remote access via https://rds.id.nithesh.acca and certificates..."
119 $RdsService = Get-Service -Name "RDS"
120 if ($RdsService) {
121     Write-Log "RDS service: Available"
122 } else {
123     Write-Log "RDS service: Not Available"
124 }
125
126 }
127
128 Completed
```

```
Administrator: Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help

Script_1.ps1 Script_2.ps1 Script_3.ps1 X

64 foreach ($hop in $TracerouteResult.TraceRoute) {
65     Write-Log "$hop"
66 }
67 if ($TracerouteResult.TraceRoute -contains $VpnGateway) {
68     Write-Log "VPN usage confirmed: Path includes VPN gateway ($VpnGateway)"
69 } else {
70     Write-Log "WARNING: VPN usage not confirmed: Path does not include VPN gateway ($VpnGateway)"
71 }
72 } else {
73     Write-Log "Traceroute to $CardiffServer: Not Available"
74 }
75
76 # 3. Check Disk Drive (D:\SwanseaFiles) Availability
77 Write-Log "Checking D:\SwanseaFiles share availability for Swansea and Cardiff networks..."
78 $Share = Get-WebShare -Name "SwanseaFiles"
79 if ($Share) {
80     $ShareAccess = Get-SmbShareAccess -Name "SwanseaFiles"
81     $DomainUsersAccess = $ShareAccess | where-Object { $_.AccountName -like "$($DomainName)\Domain Users*" -and $_.AccessRight -eq "Read" }
82     if ($DomainUsersAccess) {
83         Write-Log "D:\SwanseaFiles share: Available (Read access for Domain Users)"
84     } else {
85         Write-Log "D:\SwanseaFiles share: Not Available (No read access for Domain Users)"
86     }
87 } else {
88     Write-Log "D:\SwanseaFiles share: Not Available"
89 }
90
91 # 4. Check IIS Server with Internal authentication
92 Write-Log "Checking IIS server with internal authentication for https://webs.$($DomainName)..."
93 $IISService = Get-Service -Name "IIS"
94 if ($IISService -and $IISService.Status -eq "Running") {
95     Write-Log "IIS service: Available"
96     $WebSite = Get-WebSite -Name "webs.$($DomainName)"
97     if ($WebSite) {
98         $Auth = Get-WebConfigurationProperty -Filter "/system.webServer/security/authentication/windowsAuthentication" -Name enabled -PSPath "IIS::Sites\webs.$($DomainName)"
99         if ($Auth) {
100             Write-Log "IIS website https://webs.$($DomainName) with Windows authentication: Available"
101             try {
102                 $Response = Invoke-WebRequest -Uri "https://webs.$($DomainName)" -UseDefaultCredentials
103                 if ($Response.StatusCode -like "2*" -and $Response.Headers["Server"] -like "Microsoft-IIS/10.0") {
104                     Write-Log "IIS website content: 'Welcome to Swansea Website': Available"
105                 } else {
106                     Write-Log "IIS website content: 'Welcome to Swansea Website': Not Available"
107                 }
108             } catch {
109                 Write-Log "Website access https://webs.$($DomainName): Not Available"
110             }
111         } else {
112             Write-Log "IIS website https://webs.$($DomainName) with Windows authentication: Not Available"
113         }
114     } else {
115         Write-Log "IIS website webs.$($DomainName): Not Available"
116     }
117 } else {
118     Write-Log "IIS service: Not Available"
119 }
120
121 # 5. Check S-DC Remote Access via https://rds.id.nithesh.acca and Certificates
122 Write-Log "Checking S-DC remote access via https://rds.id.nithesh.acca and certificates..."
123 $RdsService = Get-Service -Name "RDS"
124 if ($RdsService) {
125     Write-Log "RDS service: Available"
126 } else {
127     Write-Log "RDS service: Not Available"
128 }
129
130 }
```