# Windows Server Deployment Guide on AWS Cloud

Sou Chanrojame, Orn Pheakdey, Long Neron, Then Sivthean, Le Sreyma

November 30, 2025

**Abstract**

This document provides a comprehensive step-by-step guide for deploying various Windows Server roles and services on the AWS Cloud infrastructure, including configuration details for EC2 instances, security groups, and storage optimization.

## Contents

**Prerequisites** **4**

**1 File Server** **5**
1.1 AWS Configuration . . . . . 5
1.2 Implementation Steps . . . . . 5
1.3 Best Practices . . . . . 5

**2 Proxy Server (Caching, Control Access)** **6**
2.1 AWS Configuration . . . . . 6
2.2 Implementation Steps . . . . . 6
2.3 Best Practices . . . . . 6

**3 DNS Server** **7**
3.1 AWS Configuration . . . . . 7
3.2 Implementation Steps . . . . . 7
3.3 Best Practices . . . . . 7

**4 DHCP Server** **8**
4.1 AWS Configuration . . . . . 8
4.2 Implementation Steps . . . . . 8
4.3 Best Practices . . . . . 8

**5 VPN Server** **9**
5.1 AWS Configuration . . . . . 9
5.2 Implementation Steps . . . . . 9
5.3 Alternative: AWS Client VPN . . . . . 9
5.4 Best Practices . . . . . 9

**6 Terminal Server (Thin Clients)** **10**
6.1 AWS Configuration . . . . . 10
6.2 Implementation Steps . . . . . 10
6.3 Best Practices . . . . . 10

**7 Web Server** 11
7.1 AWS Configuration . . . 11
7.2 Implementation Steps . . . 11
7.3 Best Practices . . . 11

**8 Mail Server** 12
8.1 AWS Configuration . . . 12
8.2 Implementation Steps . . . 12
8.3 Alternative: Amazon SES . . . 12
8.4 Best Practices . . . 13

**9 Database Server** 14
9.1 AWS Configuration . . . 14
9.2 Implementation Steps . . . 14
9.2.1 SQL Server . . . 14
9.2.2 PostgreSQL . . . 15
9.2.3 MongoDB . . . 15
9.3 Alternative: Amazon RDS . . . 15
9.4 Best Practices . . . 16

**10 Backup Server** 17
10.1 AWS Configuration . . . 17
10.2 Implementation Steps . . . 17
10.3 Best Practices . . . 17

**11 Load Balancing** 19
11.1 AWS Configuration . . . 19
11.2 Implementation Steps . . . 19
11.3 Best Practices . . . 20

**12 Failover Cluster** 21
12.1 AWS Configuration . . . 21
12.2 Implementation Steps . . . 21
12.3 Common Cluster Types in AWS . . . 21
12.4 Best Practices . . . 22

**13 FTP Server** 23
13.1 AWS Configuration . . . 23
13.2 Implementation Steps . . . 23
13.3 Alternative: AWS Transfer Family . . . 24
13.4 Best Practices . . . 24

**14 Container (Docker)** 25
14.1 AWS Configuration . . . 25
14.2 Implementation Steps . . . 25
14.3 Alternative: Amazon ECS for Windows Containers . . . 26
14.3.1 ECS Windows Container Setup . . . 26
14.4 Best Practices . . . 27

**15 Domain Controller** **28**
15.1 AWS Configuration . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
15.2 Installation Steps . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
15.3 Installation Steps - PowerShell . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
15.4 Security Best Practices . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 31

## Prerequisites

### AWS Account Setup

- Active AWS account with appropriate permissions
- VPC configured with public and private subnets
- Security groups properly configured
- Key pairs created for RDP access
- IAM roles for EC2 instances

### General Windows Server Launch Steps

1. Navigate to EC2 Dashboard in AWS Console
2. Click “Launch Instance”
3. Select Windows Server AMI (2019/2022 recommended)
4. Choose instance type based on workload
5. Configure instance details (VPC, subnet, IAM role)
6. Add storage as needed
7. Configure security groups
8. Review and launch with key pair

# 1 File Server

## 1.1 AWS Configuration

**Instance Type:** t3.medium or larger
**Storage:** EBS volumes with provisioned IOPS for performance
**Security Group Ports:** 445 (SMB), 139 (NetBIOS), 3389 (RDP)

## 1.2 Implementation Steps

1. **Launch Windows Server EC2 Instance**

   - Select Windows Server 2022 Datacenter
   - Attach additional EBS volumes for file storage

2. **Install File Server Role**

```
Install-WindowsFeature -Name FS-FileServer -IncludeManagementTools
Install-WindowsFeature -Name FS-DFS-Namespace, FS-DFS-Replication
```

3. **Configure Storage**

   - Initialize and format additional EBS volumes
   - Create shared folders

```
New-SmbShare -Name "SharedFiles" -Path "D:\Shares" -FullAccess "Domain\Admins" -
    ReadAccess "Domain\Users"
```

4. **Enable Shadow Copies**

```
Enable-ComputerRestore -Drive "D:\"
vssadmin resize shadowstorage /for=D: /on=D: /maxsize=20%
```

5. **Configure AWS Backup**

   - Create backup plan for EBS volumes
   - Set retention policies

## 1.3 Best Practices

- Use AWS Storage Gateway for hybrid scenarios
- Implement Amazon FSx for Windows File Server for managed solution
- Enable encryption at rest using AWS KMS
- Configure NTFS permissions and share permissions

# 2 Proxy Server (Caching, Control Access)

## 2.1 AWS Configuration

**Instance Type:** t3.medium
**Security Group Ports:** 8080, 3128 (proxy), 3389 (RDP)

## 2.2 Implementation Steps

1. **Launch Windows Server Instance**
2. **Install Proxy Server Software**
   - **Option A: Windows Server with WinGate**
     Download and install WinGate; Configure proxy settings.
   - **Option B: Squid for Windows**
     Download Squid for Windows; Install and configure `squid.conf`.
3. **Configure Proxy Settings**

```
# Example configuration for basic proxy
netsh winhttp set proxy proxy-server="localhost:8080" bypass-list="*.local"
```

4. **Set Up Caching**
   - Configure cache directory on separate EBS volume
   - Set cache size limits
   - Define cache policies
5. **Access Control**
   - Configure authentication (AD integration)
   - Set up URL filtering rules
   - Implement blacklists/whitelists
6. **Configure AWS Security Group**
   - Allow inbound traffic on proxy port from specific CIDR blocks
   - Restrict outbound traffic as needed

## 2.3 Best Practices

- Use AWS Network Firewall for additional security
- Consider AWS Global Accelerator for multiple regions
- Monitor with CloudWatch metrics
- Use ALB/NLB for proxy clustering

# 3 DNS Server

## 3.1 AWS Configuration

**Instance Type:** t3.small
**Security Group Ports:** 53 (TCP/UDP), 3389 (RDP)

## 3.2 Implementation Steps

1. **Launch Windows Server Instance**
   - Place in private subnet for internal DNS
2. **Install DNS Server Role**

```
Install-WindowsFeature -Name DNS -IncludeManagementTools
```

3. **Configure DNS Zones**

```
# Create Primary Zone
Add-DnsServerPrimaryZone -Name "yourdomain.local" -ReplicationScope "Forest" -
    PassThru

# Create Reverse Lookup Zone
Add-DnsServerPrimaryZone -NetworkID "10.0.0.0/16" -ReplicationScope "Forest"
```

4. **Configure Forwarders**

```
# Use AWS DNS or external DNS
Add-DnsServerForwarder -IPAddress "8.8.8.8", "8.8.4.4"
```

5. **Integrate with AWS Route 53**
   - Create Route 53 Resolver endpoints
   - Configure conditional forwarding for AWS resources
6. **Configure DHCP Option Sets**
   - Update VPC DHCP options to point to DNS server

## 3.3 Best Practices

- Deploy multiple DNS servers for redundancy
- Use Route 53 for public DNS records
- Enable DNS logging and monitoring
- Implement DNSSEC for security

# 4 DHCP Server

## 4.1 AWS Configuration

**Instance Type:** t3.small
**Note:** AWS VPC provides DHCP by default; custom DHCP server is optional.

## 4.2 Implementation Steps

1. **Launch Windows Server Instance**

2. **Install DHCP Server Role**

```
Install-WindowsFeature -Name DHCP -IncludeManagementTools
Add-DhcpServerInDC -DnsName "dhcp.yourdomain.local"
```

3. **Configure DHCP Scope**

```
Add-DhcpServerv4Scope -Name "Internal Network" -StartRange 10.0.1.100 -EndRange
    10.0.1.200 -SubnetMask 255.255.255.0

Set-DhcpServerv4OptionValue -ScopeId 10.0.1.0 -Router 10.0.1.1
Set-DhcpServerv4OptionValue -ScopeId 10.0.1.0 -DnsServer 10.0.1.10
```

4. **Configure Reservations**

```
Add-DhcpServerv4Reservation -ScopeId 10.0.1.0 -IPAddress 10.0.1.50 -ClientId "
    00-11-22-33-44-55" -Description "Print Server"
```

5. **Authorize DHCP Server**

```
Add-DhcpServerInDC -DnsName "dhcp.yourdomain.local" -IPAddress 10.0.1.10
```

## 4.3 Best Practices

- Consider using AWS-provided DHCP for simplicity
- Deploy DHCP failover for redundancy
- Use DHCP policies for different device types
- Monitor DHCP lease utilization

# 5 VPN Server

## 5.1 AWS Configuration

**Instance Type:** t3.small to t3.medium
**Security Group Ports:** 1723 (PPTP), 1701 (L2TP), 500/4500 (IPSec), 443 (SSTP)
**Elastic IP:** Required for consistent endpoint

## 5.2 Implementation Steps

1. **Launch Windows Server Instance with Elastic IP**
2. **Install Remote Access Role**

```
Install-WindowsFeature -Name RemoteAccess -IncludeManagementTools
Install-WindowsFeature -Name DirectAccess-VPN -IncludeManagementTools
Install-WindowsFeature -Name Routing -IncludeManagementTools
```

3. **Configure VPN Server**

```
Install-RemoteAccess -VpnType Vpn
```

4. **Configure VPN Protocols**
   - Enable SSTP, L2TP/IPSec, or IKEv2
   - Configure authentication methods (RADIUS, certificates)
5. **Set Up IP Address Assignment**

```
Set-VpnServerConfiguration -TunnelType SSTP -PassThru
```

6. **Configure Routing**
   - Enable NAT for VPN clients
   - Configure routing tables

## 5.3 Alternative: AWS Client VPN

Consider using AWS Client VPN for managed VPN service with better scalability and integration.

## 5.4 Best Practices

- Use certificate-based authentication
- Integrate with AWS Directory Service
- Monitor connections with CloudWatch
- Consider AWS Site-to-Site VPN for office connectivity

# 6 Terminal Server (Thin Clients)

## 6.1 AWS Configuration

**Instance Type:** t3.xlarge or larger (based on user count)
**Security Group Ports:** 3389 (RDP), 3391 (RD Gateway)

## 6.2 Implementation Steps

1. **Launch Windows Server Instance**
   - Size appropriately for concurrent users (2 vCPU + 4GB RAM per 5-10 users)
2. **Install RDS Roles**
```
Install-WindowsFeature -Name RDS-RD-Server -IncludeManagementTools
Install-WindowsFeature -Name RDS-Connection-Broker -IncludeManagementTools
Install-WindowsFeature -Name RDS-Web-Access -IncludeManagementTools
Install-WindowsFeature -Name RDS-Gateway -IncludeManagementTools
Install-WindowsFeature -Name RDS-Licensing -IncludeManagementTools
```
3. **Configure RDS Deployment**
   - Use Server Manager to create RDS deployment
   - Add RD Session Host
   - Configure RD Gateway for external access
4. **Install RDS CALs**
   - Install RDS License Server
   - Activate and install Client Access Licenses
5. **Configure Session Collections**
```
New-RDSessionCollection -CollectionName "Production" -SessionHost "rdsh01.yourdomain
    .local" -ConnectionBroker "rdcb.yourdomain.local"
```
6. **Set Up RemoteApp**
```
New-RDRemoteApp -CollectionName "Production" -DisplayName "Microsoft Word" -FilePath
     "C:\Program Files\Microsoft Office\root\Office16\WINWORD.EXE"
```

## 6.3 Best Practices

- Use RD Gateway with SSL certificates
- Deploy multiple Session Hosts with load balancing
- Use FSLogix for user profile management
- Store user data on separate file server
- Consider Amazon WorkSpaces for managed VDI

# 7 Web Server

## 7.1 AWS Configuration

**Instance Type:** t3.medium
**Security Group Ports:** 80 (HTTP), 443 (HTTPS), 3389 (RDP)
**Load Balancer:** Application Load Balancer recommended

## 7.2 Implementation Steps

1. **Launch Windows Server Instance**
2. **Install IIS Role**

```
Install-WindowsFeature -Name Web-Server -IncludeManagementTools
Install-WindowsFeature -Name Web-Asp-Net45, Web-Net-Ext45
Install-WindowsFeature -Name Web-Mgmt-Console
```

3. **Configure IIS**

```
# Create new website
New-Website -Name "MyWebsite" -Port 80 -PhysicalPath "C:\inetpub\MyWebsite" -
    ApplicationPool "DefaultAppPool"

# Create application pool
New-WebAppPool -Name "MyAppPool"
Set-ItemProperty IIS:\AppPools\MyAppPool -name "managedRuntimeVersion" -value "v4.0"
```

4. **Install SSL Certificate**

   - Request certificate from AWS Certificate Manager
   - Import certificate to IIS

```
New-WebBinding -Name "MyWebsite" -Protocol "https" -Port 443 -SslFlags 0
```

5. **Configure Application Settings**

   - Set up .NET Framework or .NET Core
   - Configure connection strings
   - Set permissions for application folders

6. **Set Up Application Load Balancer**

   - Create target group with health checks
   - Register EC2 instances
   - Configure listener rules

## 7.3 Best Practices

- Use AWS Certificate Manager for SSL certificates
- Enable CloudFront for CDN
- Configure auto-scaling for traffic spikes
- Use Amazon RDS instead of local database
- Enable AWS WAF for security

# 8 Mail Server

## 8.1 AWS Configuration

**Instance Type:** t3.medium
**Security Group Ports:** 25 (SMTP), 110 (POP3), 143 (IMAP), 587 (Submission), 993 (IMAPS), 995 (POP3S)
**Elastic IP:** Required
**Note:** AWS blocks port 25 by default; request removal.

## 8.2 Implementation Steps

1. **Request Port 25 Unblocking**
   - Submit request to AWS Support
   - Provide reverse DNS setup
2. **Launch Windows Server Instance with Elastic IP**
3. **Install SMTP Server**
   ```
   Install-WindowsFeature -Name SMTP-Server -IncludeManagementTools
   ```
4. **Install Third-Party Mail Server**
   - **Option A: hMailServer (Free)**
     Download and install hMailServer; Configure domains and accounts; Set up SSL/TLS certificates.
   - **Option B: Microsoft Exchange Server**
     More complex but full-featured; Install prerequisites; Install Exchange Server; Configure mailbox databases.
5. **Configure DNS Records**
   - MX records pointing to Elastic IP
   - SPF, DKIM, and DMARC records
   - Reverse DNS (PTR) record
6. **Configure Security**
   - Enable spam filtering
   - Configure antivirus scanning
   - Set up SSL/TLS encryption
   - Configure relay restrictions

## 8.3 Alternative: Amazon SES

Consider using Amazon Simple Email Service (SES) for sending emails, which provides better deliverability and doesn't require managing infrastructure.

### 8.4 Best Practices

- Use Amazon WorkMail for managed email service
- Implement proper email security (SPF, DKIM, DMARC)
- Configure backup MX records
- Monitor email queues and logs
- Use SES for transactional emails

# 9 Database Server

## 9.1 AWS Configuration

**Instance Type:** r5.large or larger (memory-optimized)
**Storage:** EBS with provisioned IOPS or io2
**Security Group Ports:**

- MongoDB: 27017
- Oracle: 1521
- SQL Server: 1433
- PostgreSQL: 5432

## 9.2 Implementation Steps

### 9.2.1 SQL Server

1. **Launch Windows Server Instance**
   - Use memory-optimized instance type
   - Attach high-performance EBS volumes
2. **Install SQL Server**
   - Download SQL Server (Developer/Standard/Enterprise)
   - Run setup.exe

```
# Silent installation example
Setup.exe /Q /ACTION=Install /FEATURES=SQLEngine /INSTANCENAME=MSSQLSERVER /
    SQLSYSADMINACCOUNTS="DOMAIN\SQLAdmins" /AGTSVCACCOUNT="NT AUTHORITY\SYSTEM" /
    SQLSVCACCOUNT="NT AUTHORITY\SYSTEM"
```

3. **Configure SQL Server**

```
-- Enable remote connections
EXEC sys.sp_configure 'remote access', 1;
RECONFIGURE;

-- Configure max memory
EXEC sys.sp_configure 'max server memory (MB)', 8192;
RECONFIGURE;
```

4. **Set Up Backups**
   - Configure SQL Server backup to S3
   - Use SQL Server native backup to S3

```
BACKUP DATABASE [MyDB] TO URL = 's3://my-bucket/backups/MyDB.bak'
```

### 9.2.2 PostgreSQL

1. **Install PostgreSQL**

   - Download PostgreSQL installer for Windows
   - Run installation wizard

2. **Configure PostgreSQL**

```
# Edit postgresql.conf
listen_addresses = '*'
max_connections = 100
shared_buffers = 2GB

# Edit pg_hba.conf for authentication
host all all 0.0.0.0/0 md5
```

3. **Create Database**

```
CREATE DATABASE myapp;
CREATE USER appuser WITH ENCRYPTED PASSWORD 'password';
GRANT ALL PRIVILEGES ON DATABASE myapp TO appuser;
```

### 9.2.3 MongoDB

1. **Install MongoDB**

   - Download MongoDB Community Server for Windows
   - Install as Windows Service

2. **Configure MongoDB**

```
# Edit mongod.cfg
net:
  port: 27017
  bindIp: 0.0.0.0
security:
  authorization: enabled
storage:
  dbPath: D:\MongoDB\data
```

3. **Create Admin User**

```
use admin
db.createUser({
  user: "admin",
  pwd: "password",
  roles: [ { role: "root", db: "admin" } ]
})
```

## 9.3 Alternative: Amazon RDS

Consider using Amazon RDS for SQL Server, PostgreSQL, or Oracle for fully managed database service with automated backups, patching, and high availability.

## 9.4 Best Practices

- Use Amazon RDS for managed database services
- Enable automated backups
- Use Multi-AZ deployments for high availability
- Store database files on separate EBS volumes
- Enable encryption at rest
- Use IAM database authentication where possible
- Monitor with CloudWatch and Performance Insights

# 10 Backup Server

## 10.1 AWS Configuration

**Instance Type:** t3.medium
**Storage:** Large EBS volumes or S3 integration
**IAM Role:** Permissions for S3, EBS snapshots

## 10.2 Implementation Steps

1. **Launch Windows Server Instance**
2. **Install Windows Server Backup**

```
Install-WindowsFeature -Name Windows-Server-Backup -IncludeManagementTools
```

3. **Configure AWS Backup**
   - Set up AWS Backup service
   - Create backup plans
   - Assign resources to backup plans
4. **Install Third-Party Backup Software**
   - **Option A: Veeam Backup**
     Download Veeam Backup & Replication; Install and configure; Set up backup jobs to S3.
   - **Option B: Windows Server Backup to S3**

```
# Create backup policy
$Policy = New-WBPolicy
$Target = New-WBBackupTarget -VolumePath "D:"
Add-WBBackupTarget -Policy $Policy -Target $Target
Add-WBVolume -Policy $Policy -Volume (Get-WBVolume -VolumePath "C:")
Set-WBSchedule -Policy $Policy -Schedule 02:00
Set-WBPolicy -Policy $Policy
```

5. **Configure S3 Lifecycle Policies**
   - Transition to S3 Glacier for long-term retention
   - Set expiration policies
6. **Set Up EBS Snapshot Automation**

```
# Using AWS PowerShell
New-EC2Snapshot -VolumeId vol-12345678 -Description "Daily Backup"
```

## 10.3 Best Practices

- Use AWS Backup for centralized backup management
- Store backups in S3 with versioning enabled
- Implement 3-2-1 backup strategy

- Test backup restoration regularly
- Use S3 Glacier for long-term archival
- Enable cross-region backup replication

# 11 Load Balancing

## 11.1 AWS Configuration

**Service:** Application Load Balancer (ALB) or Network Load Balancer (NLB)
**Target Group:** Multiple Windows Server instances

## 11.2 Implementation Steps

1. **Launch Multiple Windows Server Instances**

   - Deploy identical servers in different availability zones
   - Install and configure web application on all instances

2. **Create Target Group**

   - Navigate to EC2 > Target Groups
   - Create target group with health check settings

   ```
   Protocol: HTTP/HTTPS
   Port: 80/443
   Health Check Path: /health
   Health Check Interval: 30 seconds
   Healthy Threshold: 2
   Unhealthy Threshold: 2
   ```

3. **Create Application Load Balancer**

   - Choose ALB for HTTP/HTTPS traffic
   - Select availability zones
   - Configure security groups
   - Add listener rules
   - Register target group

4. **Configure Session Persistence**

   - Enable sticky sessions if needed
   - Configure duration

5. **Set Up Auto Scaling Group**

   ```
   # Using AWS CLI or CloudFormation
   # Define launch template
   # Create auto-scaling group
   # Configure scaling policies
   ```

6. **Install and Configure IIS ARR (Alternative)**

   ```
   # For Windows-based load balancing
   Install-WindowsFeature Web-Server -IncludeManagementTools
   # Install Application Request Routing
   # Configure server farms
   ```

### 11.3 Best Practices

- Use ALB for HTTP/HTTPS traffic
- Use NLB for TCP/UDP traffic or ultra-low latency
- Deploy instances across multiple availability zones
- Configure proper health checks
- Enable access logs for troubleshooting
- Use CloudWatch for monitoring
- Implement auto-scaling based on metrics

# 12 Failover Cluster

## 12.1 AWS Configuration

**Instance Type:** r5.xlarge or larger
**Storage:** Shared storage using FSx for Windows or S3
**Network:** Placement groups for low latency
**Security Group:** Allow cluster communication ports

## 12.2 Implementation Steps

1. **Launch Multiple Windows Server Instances**
   - Deploy in same VPC, different availability zones
   - Use placement group for low latency
2. **Install Failover Clustering Feature**
```
Install-WindowsFeature -Name Failover-Clustering -IncludeManagementTools
```
3. **Configure Shared Storage**
   - **Option A: Amazon FSx for Windows File Server**
     Create FSx file system; Mount on all cluster nodes.
   - **Option B: EBS Multi-Attach (io2 only)**
     Attach same EBS volume to multiple instances; Initialize as cluster shared volume.
4. **Create Failover Cluster**
```
# Validate cluster configuration
Test-Cluster -Node "Node1", "Node2"

# Create cluster
New-Cluster -Name "MyCluster" -Node "Node1", "Node2" -StaticAddress "10.0.1.100" -
    NoStorage
```
5. **Configure Cluster Quorum**
```
Set-ClusterQuorum -NodeAndFileShareMajority "\\FSx\Witness"
```
6. **Add Clustered Role**
```
# For SQL Server
Add-ClusterServerRole -Name "SQL-Cluster" -Storage "Cluster Disk 1"
```
7. **Configure Secondary Private IP**
   - Assign secondary private IP to ENI
   - Configure in cluster as virtual IP

## 12.3 Common Cluster Types in AWS

**SQL Server Failover Cluster** Use FSx for shared storage; Configure SQL Server on cluster nodes; Set up availability group for database replication.

**File Server Cluster** Use FSx or S3 for storage; Configure highly available file shares.

## 12.4 Best Practices

- Use Amazon FSx for Windows File Server for shared storage
- Deploy cluster nodes in different availability zones
- Use Elastic IP or Network Load Balancer for client access
- Monitor cluster health with CloudWatch
- Regular testing of failover scenarios
- Consider Amazon RDS Multi-AZ for database clustering

# 13 FTP Server

## 13.1 AWS Configuration

**Instance Type:** t3.small to t3.medium
**Security Group Ports:** 21 (FTP Control), 20 (FTP Data), 990 (FTPS), Range for Passive Mode (e.g., 50000-50100)
**Elastic IP:** Required for consistent access

## 13.2 Implementation Steps

1. **Launch Windows Server Instance with Elastic IP**

2. **Install FTP Server Role**

```
Install-WindowsFeature -Name Web-Ftp-Server -IncludeManagementTools
Install-WindowsFeature -Name Web-Ftp-Service
```

3. **Configure FTP Site**

```
# Create FTP site
New-WebFtpSite -Name "FTP Site" -Port 21 -PhysicalPath "D:\FTP"

# Configure authentication
Set-WebConfigurationProperty -Filter /system.ftpServer/security/authentication/
    basicAuthentication -PSPath IIS:\ -Location "FTP Site" -Name enabled -Value
    $true
```

4. **Configure Passive Mode**

```
# Set passive port range
Set-WebConfigurationProperty -Filter /system.ftpServer/firewallSupport -PSPath IIS:\
     -Name lowDataChannelPort -Value 50000
Set-WebConfigurationProperty -Filter /system.ftpServer/firewallSupport -PSPath IIS:\
     -Name highDataChannelPort -Value 50100

# Set external IP
Set-WebConfigurationProperty -Filter /system.ftpServer/firewallSupport -PSPath IIS:\
     -Name externalIp4Address -Value "YOUR_ELASTIC_IP"
```

5. **Enable FTPS (FTP over SSL)**

```
# Import SSL certificate
$cert = New-SelfSignedCertificate -DnsName "ftp.yourdomain.com" -CertStoreLocation
    cert:\LocalMachine\My

# Bind certificate to FTP site
Set-WebConfigurationProperty -Filter /system.ftpServer/security/ssl -PSPath IIS:\ -
    Location "FTP Site" -Name serverCertHash -Value $cert.Thumbprint
Set-WebConfigurationProperty -Filter /system.ftpServer/security/ssl -PSPath IIS:\ -
    Location "FTP Site" -Name ssl128 -Value $true
```

6. **Configure User Access**

```
# Create FTP user
New-LocalUser -Name "ftpuser" -Password (ConvertTo-SecureString "Password123!" -
    AsPlainText -Force)

# Set folder permissions
```

```
$acl = Get-Acl "D:\FTP"
$permission = "ftpuser","FullControl","Allow"
$accessRule = New-Object System.Security.AccessControl.FileSystemAccessRule
    $permission
$acl.SetAccessRule($accessRule)
Set-Acl "D:\FTP" $acl
```

7. **Configure Security Group**
    - Allow port 21 (control)
    - Allow passive port range (50000-50100)
    - Restrict source IPs if possible

## 13.3 Alternative: AWS Transfer Family

Consider using AWS Transfer Family (SFTP, FTPS, FTP) for fully managed file transfer service with S3 backend.

## 13.4 Best Practices

- Use FTPS or SFTP instead of plain FTP
- Use AWS Transfer Family for managed solution
- Store files on S3 via AWS Transfer Family
- Limit source IP addresses in security groups
- Use separate EBS volume for FTP data
- Monitor with CloudWatch logs
- Regular security audits

# 14 Container (Docker)

## 14.1 AWS Configuration

**Instance Type:** t3.medium or larger
**Operating System:** Windows Server 2019/2022 with Containers
**Security Group Ports:** Custom ports based on containerized applications

## 14.2 Implementation Steps

1. **Launch Windows Server Instance**
   - Choose Windows Server 2019/2022
   - Select version with container support
2. **Install Docker**

```
# Install Docker provider
Install-Module -Name DockerMsftProvider -Repository PSGallery -Force

# Install Docker
Install-Package -Name docker -ProviderName DockerMsftProvider -Force

# Restart computer
Restart-Computer -Force
```

3. **Verify Docker Installation**

```
docker version
docker info
```

4. **Pull Windows Container Images**

```
# Pull Windows Server Core base image
docker pull mcr.microsoft.com/windows/servercore:ltsc2022

# Pull .NET Framework image
docker pull mcr.microsoft.com/dotnet/framework/aspnet:4.8
```

5. **Create Dockerfile**

```
FROM mcr.microsoft.com/dotnet/framework/aspnet:4.8
WORKDIR /inetpub/wwwroot
COPY ./app .
EXPOSE 80
```

6. **Build and Run Container**

```
# Build image
docker build -t mywebapp:v1 .

# Run container
docker run -d -p 80:80 --name webapp mywebapp:v1

# View running containers
docker ps
```

7. **Push to Amazon ECR**

```
# Authenticate to ECR
aws ecr get-login-password --region us-east-1 | docker login --username AWS --
    password-stdin ACCOUNT_ID.dkr.ecr.us-east-1.amazonaws.com

# Tag image
docker tag mywebapp:v1 ACCOUNT_ID.dkr.ecr.us-east-1.amazonaws.com/mywebapp:v1

# Push image
docker push ACCOUNT_ID.dkr.ecr.us-east-1.amazonaws.com/mywebapp:v1
```

## 14.3 Alternative: Amazon ECS for Windows Containers

Use Amazon Elastic Container Service (ECS) or Amazon Elastic Kubernetes Service (EKS) with Windows support for orchestrated container deployments.

### 14.3.1 ECS Windows Container Setup

1. **Create ECS Cluster**
   - Choose EC2 launch type with Windows AMI
   - Or use Fargate for Windows (when available)
2. **Create Task Definition**

```
{
  "family": "windows-webapp",
  "containerDefinitions": [
    {
      "name": "webapp",
      "image": "ACCOUNT_ID.dkr.ecr.us-east-1.amazonaws.com/mywebapp:v1",
      "memory": 2048,
      "cpu": 1024,
      "portMappings": [
        {
          "containerPort": 80,
          "protocol": "tcp"
        }
      ]
    }
  ],
  "requiresCompatibilities": ["EC2"],
  "networkMode": "awsvpc",
  "runtimePlatform": {
    "operatingSystemFamily": "WINDOWS_SERVER_2022_CORE"
  }
}
```

3. **Create ECS Service**
   - Deploy task definition
   - Configure load balancer
   - Set desired task count

## 14.4 Best Practices

- Use Amazon ECS or EKS for production container orchestration
- Store images in Amazon ECR
- Use Windows Server Core or Nano Server base images
- Implement CI/CD with AWS CodePipeline
- Monitor containers with CloudWatch Container Insights
- Use Task roles for AWS service access
- Regular image security scanning

# 15 Domain Controller

## 15.1 AWS Configuration

**Instance Type:** t3.medium or larger
**Security Group Ports:**

- 53 (DNS TCP/UDP)
- 88 (Kerberos)
- 135 (RPC)
- 139, 445 (SMB)
- 389, 636 (LDAP, LDAPS)
- 3268, 3269 (Global Catalog)
- 49152-65535 (Dynamic RPC)

**Storage:** Minimum 50GB SSD
**Operating System:** Windows Server 2019/2022

## 15.2 Installation Steps

1. **Prepare the Server**
   - Set a static IP address in Windows network settings
   - Configure DNS to point to itself (127.0.0.1) and a secondary DNS
   - Rename the server with a descriptive hostname
   - Ensure the system is fully updated
2. **Install Active Directory Domain Services**
   - Open Server Manager
   - Click “Add roles and features”
   - Select “Active Directory Domain Services” role
   - Include management tools when prompted
   - Complete the installation wizard
3. **Promote to Domain Controller**
   - Click the notification flag in Server Manager
   - Select “Promote this server to a domain controller”
   - Choose “Add a new forest” for a new domain or “Add a domain controller to an existing domain”
   - Specify the root domain name (e.g., company.local)
   - Set the Forest and Domain functional levels (Windows Server 2016 or higher recommended)
   - Configure DNS and Global Catalog options (typically enabled by default)
   - Set Directory Services Restore Mode (DSRM) password

- Review NetBIOS domain name
- Specify paths for AD database, log files, and SYSVOL
- Review settings and promote
- Server will restart automatically

4. **Post-Installation Configuration**

   - Verify DNS is functioning correctly
   - Create Organizational Units (OUs) for logical organization
   - Configure Group Policy Objects (GPOs) as needed
   - Set up additional domain controllers for redundancy
   - Configure Active Directory Sites and Services if multi-site
   - Implement backup strategy for system state and Active Directory
   - Configure time synchronization (PDC Emulator should sync with external source)
   - Enable and configure Active Directory Recycle Bin for easier object recovery

## 15.3 Installation Steps - PowerShell

1. **Set Static IP Address**

```powershell
# View current network adapters
Get-NetAdapter

# Set static IP (adjust values for your environment)
New-NetIPAddress -InterfaceAlias "Ethernet" -IPAddress 10.0.1.10 -PrefixLength 24 -
    DefaultGateway 10.0.1.1

# Set DNS to localhost (127.0.0.1) and secondary
Set-DnsClientServerAddress -InterfaceAlias "Ethernet" -ServerAddresses
    127.0.0.1,8.8.8.8
```

2. **Rename Computer**

```powershell
# Rename the server
Rename-Computer -NewName "DC01" -Restart
```

3. **Install AD DS Role**

```powershell
# Install Active Directory Domain Services role with management tools
Install-WindowsFeature -Name AD-Domain-Services -IncludeManagementTools

# Verify installation
Get-WindowsFeature | Where-Object {$_.Name -eq "AD-Domain-Services"}
```

4. **Promote to Domain Controller (New Forest)**

```powershell
# Import the AD DS Deployment module
Import-Module ADDSDeployment

# Create new forest and promote to DC
Install-ADDSForest `
    -DomainName "company.local" `
    -DomainNetbiosName "COMPANY" `
    -ForestMode "WinThreshold" `
    -DomainMode "WinThreshold" `
    -InstallDns:$true `
    -CreateDnsDelegation:$false `
    -DatabasePath "C:\Windows\NTDS" `
    -LogPath "C:\Windows\NTDS" `
    -SysvolPath "C:\Windows\SYSVOL" `
    -NoRebootOnCompletion:$false `
    -Force:$true
```

*Note: You'll be prompted for the SafeModeAdministratorPassword (DSRM password)*

5. **Promote Additional Domain Controller (Existing Domain)**

```
# Add DC to existing domain
Install-ADDSDomainController `
    -DomainName "company.local" `
    -InstallDns:$true `
    -Credential (Get-Credential "COMPANY\Administrator") `
    -DatabasePath "C:\Windows\NTDS" `
    -LogPath "C:\Windows\NTDS" `
    -SysvolPath "C:\Windows\SYSVOL" `
    -NoRebootOnCompletion:$false `
    -Force:$true
```

6. **Post-Installation Verification**

```
# Verify AD Web Services is running
Get-Service ADWS

# Check domain controller functionality
Get-ADDomainController

# Test AD replication (if multiple DCs)
repadmin /replsummary

# Verify DNS zones
Get-DnsServerZone

# Check SYSVOL replication
dfsrdiag replicationstate /all

# Verify FSMO roles
Get-ADDomain | Select-Object InfrastructureMaster, RIDMaster, PDCEmulator
Get-ADForest | Select-Object DomainNamingMaster, SchemaMaster
```

7. **Create Organizational Units**

```
# Create OUs for organization
New-ADOrganizationalUnit -Name "Users" -Path "DC=company,DC=local"
New-ADOrganizationalUnit -Name "Computers" -Path "DC=company,DC=local"
New-ADOrganizationalUnit -Name "Groups" -Path "DC=company,DC=local"
New-ADOrganizationalUnit -Name "Servers" -Path "DC=company,DC=local"
```

8. **Configure Active Directory Recycle Bin**

```
# Enable AD Recycle Bin (cannot be reversed)
Enable-ADOptionalFeature -Identity 'Recycle Bin Feature' `
    -Scope ForestOrConfigurationSet `
    -Target 'company.local' `
    -Confirm:$false
```

9. **Configure Time Synchronization (PDC Emulator)**

```
# Configure external time source on PDC Emulator
w32tm /config /manualpeerlist:"time.windows.com,0x8" /syncfromflags:manual /reliable
    :yes /update

# Restart Windows Time service
Restart-Service W32Time

# Force sync and check status
w32tm /resync
w32tm /query /status
```

10. **Set Password Policy**

```
# Configure default domain password policy
Set-ADDefaultDomainPasswordPolicy -Identity "company.local" `
    -ComplexityEnabled $true `
    -LockoutDuration "00:30:00" `
    -LockoutThreshold 5 `
    -MaxPasswordAge "90.00:00:00" `
    -MinPasswordAge "1.00:00:00" `
    -MinPasswordLength 12 `
    -PasswordHistoryCount 24
```

## 15.4 Security Best Practices

- Implement least privilege access for domain administrators
- Use separate administrative accounts for daily tasks vs. domain administration
- Enable and monitor security logs
- Regularly patch and update the domain controller
- Consider implementing tiered administrative model
- Use strong password policies and account lockout policies
- Utilize Advanced Threat Protection