

Deploy Siebel CRM on Azure using Siebel Installer

This whitepaper provides a comprehensive architectural roadmap for migrating Oracle Siebel CRM to Microsoft Azure Cloud. It explores the transition from traditional on-premises infrastructure to a high-availability cloud environment, detailing the installation process using the Siebel Installer and the strategic advantages of Azure's IaaS capabilities for enterprise-scale CRM.

December, 2025, Version [\[1.0\]](#)
Copyright © 2025, Oracle and/or its affiliates
Public

Purpose statement

This document provides a solution approach and is for informational purposes only. It is intended solely to provide valuable insights and guidance on the subject matter while serving as a reference for stakeholders. This document does not constitute a binding agreement or official policy.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the solution described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Table of contents

Overview	5
Prerequisites	6
Proposed Deployment Architecture diagram	5
High Level Steps for Deploying Siebel CRM on a Kubernetes Cluster	6
Installing Prerequisites	7
Step 1. Create Azure resources	7
Resource Group	7
Application Vnet	7
Kubernetes Cluster Vnet	8
Enable VNET Peering Between Application VNet and Kubernetes VNet	9
Create Network Security Groups	9
Create Virtual Machines in Application VNet	10
Setup Azure Container Registry	11
Step 2. Configure Gitlab	11
Step 3. Setting Up NFS and Private Endpoint at Azure Cloud	11
Step 4. DB setup	11
Siebel Cloud Manager Setup	12
Step 5. Siebel installer file	12
Step 6. Verifying the SCM Image and SCM Helm Chart in the Container Registry	13
Step 7. Installing SCM using Helm	14
Deploying Siebel CRM	14
Step 8. Deploying Siebel CRM using SCM	14
Troubleshooting Siebel CRM Deployment	15
Conclusion	18
Call to Action (Next Steps)	18
Appendix I: Configuring GitLab	19
Prerequisite	19
Task 1: Deploy and configure GitLab	19
Configure the Basics tab	19
Configure the Disks tab	19
Configure the Networking tab	20
Configure the Management tab	20
Configure the Advanced tab	20
Configure the Tags tab	20
Review and create the VM	21
Finish deployment	21
Summary:	21
Task 2: Change the GitLab external URL	21

Task 3: Enable HTTPS	22
Task 4: Generate a GitLab Access Token	23
Troubleshooting	24
Appendix II: Setting Up NFS and Private Endpoint at Azure Cloud	25
Overview	25
Prerequisites	25
Create a File storage account	25
Deploy an Azure VM running Linux	26
Create an NFS Azure file share	28
Add a file share to your storage account	28
Set up a private endpoint or service endpoint	29
Disable secure transfer	33
Connect to your VM	34
Mount the NFS share	35
Troubleshooting	36

List of figures and tables

Figure 1. Siebel on Azure architecture	5
Table 1. Common Issues and their solutions	15
Table 2. Prerequisites	20
Figure 2. Azure Screenshot on storage account	26
Figure 3. Azure Screenshot on Project details	27
Figure 4. Azure Screenshot on Administrator account	27
Figure 5. Azure Screenshot on Inbound port rule	28
Figure 9. Azure Screenshot on Private endpoint	30
Figure 10. Azure Screenshot on create private endpoint	31

Overview

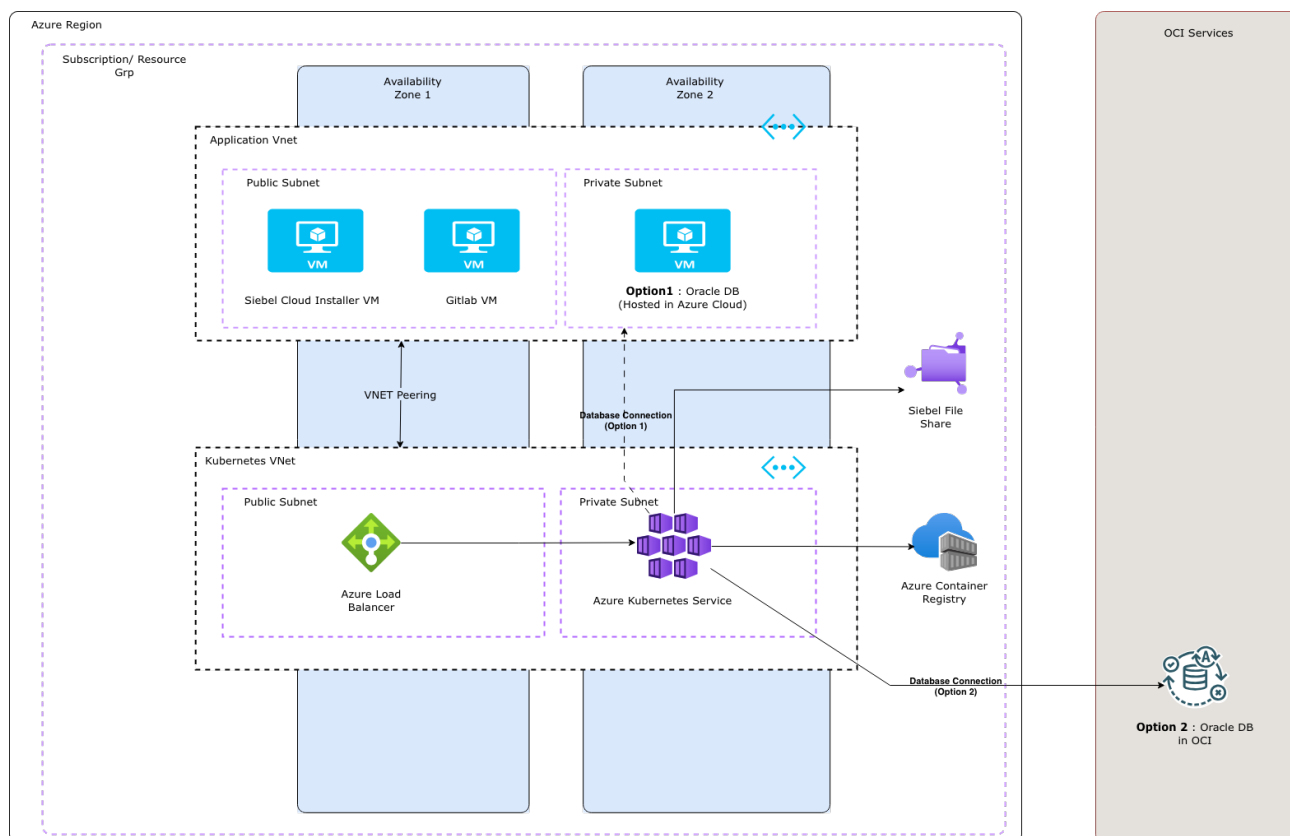
Siebel CRM (Customer Relationship Management), is a comprehensive suite of applications designed to help organizations manage and automate their customer-facing processes, spanning sales, marketing, and customer service. Recognized for its robust nature and adaptability, Siebel provides a unified, 360-degree view of the customer, enabling businesses across various industries to streamline operations, enhance customer engagement, and ultimately drive revenue growth. It is a highly configurable platform that supports diverse deployment models, accommodating the complex needs of large enterprises.

This guide outlines the steps required to deploy Siebel CRM on a Kubernetes cluster using Siebel Installer for Siebel Cloud Manager (SCM). Siebel CRM deployment on Kubernetes allows you to scale Siebel applications efficiently while utilizing cloud-native features such as automatic scaling, load balancing, and high availability.

Siebel Cloud Manager (SCM) integrates with Kubernetes to facilitate the deployment and management of Siebel CRM in a containerized environment. The Siebel Installer assists in the automation of this process.

Proposed Deployment Architecture diagram

Figure 1. Siebel on Azure architecture



Prerequisites

Before starting the deployment, ensure you have the following prerequisites in place:

- Open Container Initiative registry such as Azure Container Registry
- Kubernetes cluster and namespace such as Azure OKE Cluster
- NFS File Share in Azure Cloud
- Existing Siebel Database (We support only BYODB at this point).
- Git instance to use for Siebel CRM helm chart repository.
- Networking Configuration:
 - Proper firewall rules and network policies to ensure communication between Siebel components and Kubernetes pods.
- Linux VM version 8 or above with minimum 20 GB free disk space.
- Helm version 3.8 or later: A package that contains the resources needed to deploy an application on a Kubernetes cluster. The SCM Helm package contains the artifacts required to deploy Siebel on a Kubernetes cluster. It is also used to push the SCM package into the container registry and to deploy SCM and Siebel CRM on a Kubernetes cluster. For more information, refer the online documentation for "Installing Helm".
- Podman: An open source tool for managing containers on Linux, Windows and so on. Here, Podman is used to manage the SCM container registry. For more information, refer the online documentation for "Podman Installation".

For more information, see Prerequisites for Deploying Siebel CRM on a Kubernetes Cluster.

High Level Steps for Deploying Siebel CRM on a Kubernetes Cluster

The following are the high-level steps to deploy Siebel CRM on a CNCF certified Kubernetes cluster (these steps are described in detail later in this document):

1. Install the prerequisite tools and make the prerequisite resources available with necessary connectivity and access provisions for deploying Siebel CRM on a Kubernetes cluster.
2. Download the Siebel Installer media zip file from My Oracle Support (MOS) and extract it.
3. Run the installer and complete the setup and configuration.
4. Verify that the SCM image and SCM Helm chart are available in your container registry.
5. Install SCM using Helm.
6. Move the following container images to your container registry using SCM mirroring APIs:
 - Siebel CRM image
 - SCM utilities images
7. Verify the dependencies and prerequisites for Siebel CRM deployment.
8. Deploy Siebel CRM on a Kubernetes cluster using SCM APIs.
9. Verify Siebel CRM on the Kubernetes cluster.

Configure Network system as shown in the diagram

Installing Prerequisites

Step 1. Create Azure resources

Comprehensive deployment instructions are available via the <https://learn.microsoft.com/en-us/azure/aks/learn/quick-kubernetes-deploy-portal?tabs=azure-cli>. Readers are encouraged to refer to this official guide for further technical clarification.

Resource Group

A resource group is a container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how to add resources to resource groups based on what makes the most sense for your organization. Generally, add resources that share the same lifecycle to the same resource group so you can easily deploy, update, and delete them as a group.

The resource group stores metadata about the resources. When you specify a location for the resource group, you're specifying where that metadata is stored. For compliance reasons, you might need to ensure that your data is stored in a particular region.

To create a resource group, use `az group create`.

```
az group create --name SiebelInfraRG --location eastus
```

Using Azure Portal : <https://learn.microsoft.com/en-us/azure/azure-resource-manager/management/manage-resource-groups-portal>

Application Vnet

A virtual network is the fundamental building block for private networks in Azure. Azure Virtual Network enables Azure resources like VMs to securely communicate with each other and the internet.

Use `az network vnet create` to create a virtual network named **ApplicationVnet** with subnets

```
az network vnet create \  
  --resource-group SiebelInfraRG \  
  --name ApplicationVNet \  
  --address-prefix 10.0.0.0/16 \  
  --subnet-name AppPublicSubnet \  
  --subnet-prefix 10.0.1.0/24  
  
az network vnet subnet create \  
  --resource-group SiebelInfraRG \  
  --vnet-name ApplicationVNet \  
  --name AppPrivateSubnet \  
  --address-prefix 10.0.2.0/24
```

Using Azure Portal: <https://learn.microsoft.com/en-us/azure/virtual-network/quickstart-create-virtual-network>

Kubernetes Cluster Vnet

Azure Kubernetes Service (AKS) is a managed Kubernetes service that lets you quickly deploy and manage clusters. In this quickstart, you:

- Deploy an AKS cluster using the Azure portal.
- Run a sample multi-container application with a group of microservices and web front ends simulating a retail scenario.

Create an AKS cluster using the `az aks create` command.

```
az network vnet create \  
  --resource-group SiebelInfraRG \  
  --name KubernetesVNet \  
  --address-prefix 10.1.0.0/16 \  
  --subnet-name KubePublicSubnet \  
  --subnet-prefix 10.1.1.0/24  
  
az network vnet subnet create \  
  --resource-group SiebelInfraRG \  
  --vnet-name KubernetesVNet \  
  --name KubePrivateSubnet \  
  --address-prefix 10.1.2.0/24  
  
az aks create \  
  --resource-group SiebelInfraRG \  
  --name SiebelAKSCluster \  
  --node-count 3 \  
  --enable-addons monitoring \  
  --network-plugin azure \  
  --vnet-subnet-id $(az network vnet subnet show \  
    --resource-group SiebelInfraRG \  
    --vnet-name KubernetesVNet \  
    --name KubePrivateSubnet \  
    --query id -o tsv) \  
  --generate-ssh-keys
```

Using Azure Portal: <https://learn.microsoft.com/en-us/azure/aks/learn/quick-kubernetes-deploy-portal?tabs=azure-cli>

Enable VNET Peering Between Application VNet and Kubernetes VNet

```
az # Peer from ApplicationVNet to KubernetesVNet
az network vnet peering create \
  --name AppToKubePeering \
  --resource-group SiebelInfraRG \
  --vnet-name ApplicationVNet \
  --remote-vnet KubernetesVNet \
  --allow-vnet-access

# Peer from KubernetesVNet to ApplicationVNet
az network vnet peering create \
  --name KubeToAppPeering \
  --resource-group SiebelInfraRG \
  --vnet-name KubernetesVNet \
  --remote-vnet ApplicationVNet \
  --allow-vnet-access
```

Create Network Security Groups

You may want to create NSGs and associate them to the public/private subnets for both VNets to control traffic.

```
az network nsg create \
  --resource-group SiebelInfraRG \
  --name AppPublicNSG

az network vnet subnet update \
  --resource-group SiebelInfraRG \
  --vnet-name ApplicationVNet \
  --name AppPublicSubnet \
  --network-security-group AppPublicNSG
```

Create Virtual Machines in Application VNet

Create Siebel Cloud Installer VM (in AppPublicSubnet)

```
az vm create \  
  --resource-group SiebelInfraRG \  
  --name SiebelInstallerVM \  
  --image UbuntuLTS \  
  --vnet-name ApplicationVNet \  
  --subnet AppPublicSubnet \  
  --admin-username azureuser \  
  --generate-ssh-keys \  
  --public-ip-address ""
```

Create GitLab VM (in AppPublicSubnet)

```
az vm create \  
  --resource-group SiebelInfraRG \  
  --name GitLabVM \  
  --image UbuntuLTS \  
  --vnet-name ApplicationVNet \  
  --subnet AppPublicSubnet \  
  --admin-username azureuser \  
  --generate-ssh-keys
```

Setup Azure Container Registry

```
az acr create --resource-group SiebelInfraRG \  
  --name SiebelACR \  
  --sku Basic
```

Then connect it to AKS:

```
az aks update \  
  --name SiebelAKSCluster \  
  --resource-group SiebelInfraRG \  
  --attach-acr SiebelACR
```

Step 2. Configure Gitlab

Refer Apendix I

Step 3. Setting Up NFS and Private Endpoint at Azure Cloud

Refer Apendix II

Step 4. DB setup

BYODB with Siebel seed data

Siebel Cloud Manager Setup

Step 5. Siebel installer file

While this section provides a high-level overview, the official Siebel documentation - [Overview of Siebel Installer for SCM](#) - should be treated as the definitive source for deployment procedures and troubleshooting.

To download and run the installer:

- Download the media file from My Oracle Support (MOS). For example, p37481729_251_Linux-x86-64.zip. Note: Here, the file name is representational and will change with the build. The format of the media file name is <patch_id>_<release_tag>_<platform>.zip, where:
 - <patch_id> is the patch or the bug number. For example, 37481729.
 - <release_tag> is the SCM release version. For example, 251 for the SCM release 25.1.
 - <platform> is the platform you wish to install SCM on. For example, Linux-x86-64.
- 2. Extract the media zip file to the directory (which will be referred to as the "installation directory" in the steps later) from which you want to run the installer, as follows:

```
cd <installation_directory> unzip p37481729_251_Linux-x86-64.zip
```

The following files are extracted in the installation directory:

```
drwxrwxrwx 3 root root 3 Oct 22 10:40 ext
drwxrwxrwx 2 root root 4 Oct 22 10:41 resources
drwxrwxrwx 2 root root 2 Oct 22 10:41 logs
drwxrwxrwx 3 root root 3 Oct 22 11:13 archives
-rwxrwxrwx 3 root root 3 Oct 25 09:31 README.html
-rwxrwxrwx 1 root root 186 Oct 25 09:31 runInstaller.sh
drwxrwxrwx 3 root root 3 Oct 29 07:24 jre
-rwxrwxrwx 1 root root 53 Nov 22 06:46 installer.properties
```

Note: Make a note of this installation directory; later you'll run the SCM installation from this directory.

- (Optional) Configure the non-root user to push the images to the container registry, as follows:
Note: You must perform this step only if you want to run the installer as a non-root user.

```
sudo /sbin/usermod --add-subuids <first-last> --add-subgids <first-last>
<uid> podman system migrate /usr/bin/systemctl --user daemon-reload
```

The variables in the example have the following values:

- <uid> is the non-root user ID.
- <first-last> is the subordinate user/group IDs range that the non-root user is allowed to use. For example, 20000-265536.

- (Optional) Sign in to the container registry using Podman and Helm utilities.

Note: If you sign in to the container registry now, which is before running the installer, you need not provide the registry credentials in the SCM configuration details when you run the installer. The credentials with which you've logged in to the container registry are used to push the SCM image and SCM Helm chart to the container registry.

- Start the installer in GUI mode as follows:

```
sh runInstaller.sh
```

- Complete the installation as follows:
 1. In the Installation Location screen, click Browse to select the directory to store the SCM artifacts and click Next.
 2. In the Component Selection screen, the installation component "Siebel Cloud Manager" and configuration task "Siebel Cloud Manager Configuration" are selected by default. Click Next to continue.

Note: Don't deselect "Siebel Cloud Manager" or "Siebel Cloud Manager Configuration". If you deselect "Siebel Cloud Manager," the installer will throw an error and you'll not be able to proceed with the installation. If you deselect "Siebel Cloud Manager Configuration", the installer will not upload the SCM image and SCM Helm chart to the container registry.

3. In the Configuration Details screen, enter the container registry details required to push the SCM image and SCM Helm chart to the container registry:
 - Container registry URL: The container registry URL to push the SCM image and the SCM Helm chart to. For example, harbor-registry.corpxyz.mytenancy02phx.oraclevcn.com.
 - (Optional) Container registry URL prefix: The prefix for the container registry. When specified, the prefix is used to construct the repository path. For example, scmxyz.
 - Container registry URL UserName: The user name to access the container registry. It is optional if the installer is launched from the session in which the user has already logged in to the container registry.
 - Container registry Password: The password to access the container registry. It is optional if the installer is launched from the session in which the user has already logged in to the container registry.
4. In the Summary screen, review details and click Install.
5. Optionally, you can save the installation response messages to a response file. To create a response file:
 1. Click Save Response File.
 2. Go to the required directory.
 3. Enter the response file name.
 4. Click Save.
6. In the Installation Progress screen, as the installation progresses, appropriate messages are displayed. After you receive the message "Post Installation Configuration Complete", click Next.
7. In the Finish Installation screen, after the installation is complete, the "Installation is successful" message is displayed along with the path and name of the log file. Click Close.

Step 6. Verifying the SCM Image and SCM Helm Chart in the Container Registry

After the installer is run successfully, you must verify that the SCM image and SCM Helm chart are available in the container registry for smooth SCM installation through one of the following ways:

- Open the log file and look for the following messages:
- Uploaded image and helm chart to customer registry Post install configuration step completed for Siebel Cloud Manager Configuration Post Installation Configuration Completed.
- Sign in to the container registry and verify the image from the interface provided by the registry; for example, the GUI.

Step 7. Installing SCM using Helm

Comprehensive deployment instructions are available via the [Installing SCM using Helm](#)

Deploying Siebel CRM

Step 8. Deploying Siebel CRM using SCM

For complete procedural details and configuration parameters, refer to the official Oracle documentation:

Additional Steps

Reinstalling SCM using Helm

For complete procedural details and configuration parameters, refer to the official Oracle documentation:

[Reinstalling SCM using Helm](#)

Upgrading SCM using Helm

For complete procedural details and configuration parameters, refer to the official Oracle documentation:

[Upgrading SCM using Helm](#)

Uninstalling SCM using Helm

For complete procedural details and configuration parameters, refer to the official Oracle documentation:

[Uninstalling SCM using Helm](#)

Troubleshooting Siebel CRM Deployment

Troubleshoot Guide

Table 1. Common Issues and their solutions

Issue	Solution
Gitlab setup issue Gitlab was picking up the wrong certificates instead of one that we provided. which was causing a certificate issue.	Find the following settings and comment them out, so that GitLab doesn't pick up the wrong certificates: <pre># nginx['redirect_http_to_https'] = true # nginx['ssl_certificate'] = "/etc/gitlab/ssl/server.crt" # nginx['ssl_certificate_key'] = "/etc/gitlab/ssl/server.key"</pre>

<p>NFS version Issue NFS version 4.1 is supported in Azure. SCM supports version 3.</p>	<p>Azure NFS is of version 4.1, so we needed to add vers the option on mounting PV.</p> <p>Updated NFS version as 4.1 in pv_persst.yaml file - pv_persist.yaml</p> <p>mountOptions:</p> <ul style="list-style-type: none"> - nosuid - vers=4.1 <pre>spec: storageClassName: '' capacity: storage: {{ .Values.storage.storageSize default "100Gi" }} accessModes: - ReadWriteMany mountOptions: - nosuid - vers=4.1 persistentVolumeReclaimPolicy: {{ .Values.storage.reclaimPolicy default "Retain" }} nfs: server: {{ .Values.storage.nfsServer }} path: {{ .Values.storage.nfsPath }} readOnly: false ---</pre>
<p>Siebel payload POST was failing at helm install stage.</p>	<p>This happened due to ociDeployment (ociDeployment: "False") set to False instead of false. The parameter was case-sensitive. Correct value is "false".</p> <p>Accepted value was given as Accepted values = ["public", "false", "oc3"] in values.yaml</p>
<p>Registry prefix in mirror images API You can mirror the Siebel CRM base images that SCM uses to provision a Siebel CRM environment to a user-defined destination registry through the base/images/mirror API. The API call pulls the container images and pushes them to the user defined destination registry.</p> <p>Registry prefix should be "scm". System will look for scm_artifact under scm directory instead of registry prefix directory.</p>	<p>"registry_prefix" should be "scm". so that created files will be in below location "oraclesiebelcontainerregistry.azurecr.io/scm/busybox:latest". Registry prefix should be "scm". System will look for scm_artifact under scm directory instead of registry prefix directory.</p> <p>Sample post payload :</p> <pre>{ "destination_registry": { "registry_url": "oraclesiebelcontainerregistry.azurecr.io", "registry_user": "oraclesiebelcontainerregistry", "registry_password": "HztL8Bwf2KYzUtezS0FTLrp10f3B0avJAoh47bFrm6+ACRD6cxFX", "registry_prefix": "scm" }, "update_global_config": "true" }</pre>
<p>Siebel payload POST was failing at import_siebel_db</p>	<p>In BYOD, SCM expects the provided DB should have initial Siebel seed data.</p>

Error said Siebel user not found invalid user / password	All DB users are expected to be already created in DB.
--	--

More information also refer [Troubleshooting Siebel CRM Deployment](#)

Conclusion

Deploying Siebel CRM on Azure transforms a traditionally rigid on-premises application into a highly scalable, resilient, and modern enterprise solution. By utilizing Azure's IaaS capabilities—such as Load Balancers for high availability and Managed Disks for database performance—organizations can significantly reduce the overhead of managing hardware while gaining the ability to scale resources based on user demand.

Successfully installing Siebel using the Siebel Installer and SMC is only the first step. The true power of this deployment lies in its integration with the Azure ecosystem, allowing for enhanced security through Microsoft Entra ID (formerly Azure AD), automated backups via Azure Backup, and deep insights through Azure Monitor.

Call to Action (Next Steps)

To ensure this deployment moves successfully from the "Installed" state to a "Production-Ready" environment, please complete the following action items:

- **Security Hardening:** Review and restrict all Network Security Group (NSG) rules. Ensure that only required ports are open and that administrative access is gated through Azure Bastion.
- **Enable Monitoring:** Configure Azure Monitor and Log Analytics to track VM performance and Siebel Server log files. Set up alerts for high CPU usage or service interruptions.
- **Automated Backups:** Schedule daily snapshots of the Siebel Database VM and the Azure File Share (SFS) using Azure Backup Center.
- **Performance Tuning:** Conduct a baseline load test to ensure the Azure VM sizes (Standard D-Series or E-Series) handle your specific user concurrency without latency.

Contact us directly at siebel_coe_grp@oracle.com to discuss your implementation.

Need Help? If you encounter issues during the final configuration, please refer to the Troubleshooting or open a ticket with the Oracle support team.

Appendix I: Configuring GitLab

This section provides a step-by-step guide to setting up GitLab on Microsoft Azure. The setup includes deploying GitLab using an Azure Virtual Machine (VM) and configuring necessary network and storage settings.

Prerequisite

You need an account on Azure.

Task 1: Deploy and configure GitLab

Because GitLab is already installed in a pre-configured image, all you have to do is create a new VM:

1. Visit the GitLab offering in the marketplace
2. Select Get it now and the Create this app in Azure window opens. Select Continue.
3. Select one of the following options from the Azure portal:
 - o Select Create to create a VM from scratch.
 - o Select Start with a pre-set configuration to get started with some pre-configured options. You can modify these configurations at any time.

For the sake of this guide, let's create the VM from scratch, so select **Create**.

Be aware that Azure incurs compute charges whenever your VM is active (known as "allocated"), even if you're using free trial credits. [how to properly shutdown an Azure VM to save money](#). See the [Azure pricing calculator](#) to learn how much resources can cost.

After you create the virtual machine, use the information in the following sections to configure it.

Configure the Basics tab

The first items you need to configure are the basic settings of the underlying virtual machine:

1. Select the subscription model and a resource group (create a new one if it doesn't exist).
2. Enter a name for the VM, for example GitLab.
3. Select a region.
4. In Availability options, select Availability zone and set it to 1. Read more about the availability zones.
5. Ensure the selected image is set to GitLab - Gen1.
6. Select the VM size based on the hardware requirements. Because the minimum system requirements to run a GitLab environment for up to 500 users is covered by the D4s_v3 size, select that option.
Flex Shape Memory:
7. Set the authentication type to SSH public key.
8. Enter a user name or leave the one that is automatically created. This is the user Azure uses to connect to the VM through SSH. By default, the user has root access.
9. Determine if you want to provide your own SSH key or let Azure create one for you. For more information about how to set up SSH public keys, see SSH.

Review your entered settings, and then proceed to the Disks tab.

Configure the Disks tab

For the disks:

1. For the OS disk type, select **Premium SSD**.
2. Select the default encryption.

[Read more about the types of disks](#) that Azure provides.

Review your settings, and then proceed to the Networking tab.

Configure the Networking tab

Use this tab to define the network connectivity for your virtual machine, by configuring network interface card (NIC) settings. You can leave them at their default settings.

Azure creates a security group by default and the VM is assigned to it. The GitLab image in the marketplace has the following ports open by default:

Prerequisites Table

Table 2. Prerequisites

Port	Description
80	Enable the VM to respond to HTTP requests, allowing public access.
443	Enable our VM to respond to HTTPS requests, allowing public access.
22	Enable our VM to respond to SSH connection requests, allowing public access (with authentication) to remote terminal sessions.

If you want to change the ports or add any rules, you can do it after the VM is created by selecting Networking settings in the left sidebar, while in the VM dashboard.

OCI settings

Network Strategy : Use Existing VCN and Subnet

Existing VCN : (Choose the VCN that was previously created by the Siebel Cloud Manager stack - it should be the only option for the selected compartment)

Subnet Type : Public Subnet

Existing Subnet : (Choose the Public Subnet that was previously created by the Siebel Cloud Manager stack)

Network Security Group Configuration : Use Recommended Configuration

Configure the Management tab

Use this tab to configure monitoring and management options for your VM. You don't need to change the default settings.

Configure the Advanced tab

Use this tab to add additional configuration, agents, scripts or applications through virtual machine extensions or cloud-init. You don't need to change the default settings.

Configure the Tags tab

Use this tab to add name/value pairs that enable you to categorize resources. You don't need to change the default settings.

Review and create the VM

The final tab presents you with all your selected options, where you can review and modify your choices from the previous steps. Azure runs validation tests in the background, and if you provided all the required settings, you can create the VM.

After you select **Create**, if you had opted for Azure to create an SSH key pair for you, a prompt appears to download the private SSH key. Download the key, as it's needed to SSH into the VM.

After you download the key, the deployment begins.

Finish deployment

At this point, Azure begins to deploy your new VM. The deployment process takes a few minutes to complete. After it's complete, the new VM and its associated resources are displayed on the Azure Dashboard. Select **Go to resource** to visit the dashboard of the VM.

GitLab is now deployed and ready to be used. Before doing so, however, you need to set up the domain name and configure GitLab to use it.

Summary:

1. Log in to the Azure Portal.
2. Navigate to *Virtual Machines* and click *Create > Azure Virtual Machine*.
3. Choose an appropriate region and VM size (recommended: *Standard_D4s_v3* or higher).
4. Under *Image*, select *Ubuntu Server 20.04 LTS*.
5. Configure authentication (SSH public key or password).
6. Enable inbound ports: *HTTP (80)*, *HTTPS (443)*, and *SSH (22)*.
7. Click *Review + Create* and then *Create*.

Task 2: Change the GitLab external URL

GitLab uses `external_url` in its configuration file to set up the domain name. If you don't set this up, when you visit the Azure friendly name, the browser will redirect you to the public IP.

To set up the GitLab external URL:

1. Connect to GitLab through SSH by going to **Settings > Connect** from the VM dashboard and follow the instructions. Remember to sign in with the username and SSH key you specified when you [created the VM](#). The Azure VM domain name is the one you [set up previously](#). If you didn't set up a domain name for your VM, you can use the IP address in its place.

In the case of our example:

```
ssh -i <private key path> azureuser@<your-vm-public-ip>
```

If you need to reset your credentials, read [how to reset SSH credentials for a user on an Azure VM](#).

2. Open `/etc/gitlab/gitlab.rb` with your editor.
3. Find `external_url` and replace it with your own domain name. For the sake of this example, use the default domain name Azure sets up. Using `https` in the URL [automatically enables](#), Let's Encrypt, and sets `HTTPS` by default:

```
external_url '<your-vm-public-ip>'
```

- Find the following settings and comment them out, so that GitLab doesn't pick up the wrong certificates:

```
# nginx['redirect_http_to_https'] = true
# nginx['ssl_certificate'] = "/etc/gitlab/ssl/server.crt"
# nginx['ssl_certificate_key'] = "/etc/gitlab/ssl/server.key"
```

- Reconfigure GitLab for the changes to take effect. Run the following command every time you make changes to `/etc/gitlab/gitlab.rb`:

```
sudo gitlab-ctl reconfigure
```

- To prevent the domain name from [resetting after a reboot](#), rename the utility that Bitnami uses:

```
sudo mv /opt/bitnami/apps/gitlab/bnconfig
/opt/bitnami/apps/gitlab/bnconfig.bak
```

You can now visit GitLab with your browser at the new external URL.

Task 3: Enable HTTPS

- Create a self-signed certificates using OpenSSL. Run the following commands one by one to begin the process.

Note: Self-signed certificates are only suggested for this lab. Typically you will use Certificate Authority (CA) signed certificates. Follow [Gitlab's documentation to troubleshoot SSL issues](#)

```
mkdir -p /etc/gitlab/ssl

$ chmod 755 /etc/gitlab/ssl

$ cd /etc/gitlab/ssl

$ openssl genrsa -out rootCA.key 2048

$ openssl req -x509 -new -nodes -key rootCA.key -sha256 -days 365 -out
rootCA.crt -subj "/C=US/ST=UT/L=Lehi/O=Oracle/OU=Support/CN=localhost"

$ openssl genrsa -out {Public IP of GitLab instance}.key 2048

$ openssl req -new -key {Public IP of GitLab instance}.key -out
{Public IP of GitLab instance}.csr -subj "/CN=localhost"
```

- Create a configuration file named **device-csr.conf**

```
vi device-csr.conf
```

3. Populate this file with the following the save and close the file

```
[req]
distinguished_name = req_distinguished_name
req_extensions = v3_req
prompt = no

[req_distinguished_name]
C = US
ST = UT
L = Lehi
O = Oracle
OU = Corp
CN = localhost

[v3_req]
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
IP.1 = {Public IP of GitLab instance}
```

4. Run the following command to finally create a self-signed certificate

```
$ openssl x509 -req -in {Public IP of GitLab instance}.csr -CA
rootCA.crt -CAkey rootCA.key -CAcreateserial -out {Public IP of GitLab
```

5. Reconfigure GitLab by running the below command.

```
$ gitlab-ctl reconfigure
```

6. Copy the **rootCA.crt** from **/etc/gitlab/ssl** folder to the Siebel Cloud Manager instance's **/home/opc/certs** folder. Also copy the file to docker path

Task 4: Generate a GitLab Access Token

Access GitLab

1. Open a browser and visit <http://<your-vm-public-ip>>.
2. The initial login requires a password. Retrieve it using:

```
sudo cat /etc/gitlab/initial_root_password
```

3. Log in with username root and the retrieved password.
4. If you wish to change, the root password from the default, now is a good time. The file above will be deleted 24 hours after initial setup. Click on the user icon on the right hand side at the top left, and select Edit Profile
5. Now click the Password option on the left menu, and fill in the form to change the password, then click Save password.
6. In the left side panel, navigate to *Access Tokens* and then click Add new token on the right hand side.

7. Give the following values for the respective fields.

Name:	
	Siebel
Expires at:	
	{Give a future date not more than a year away}
Scope:	
	api

8. Scroll down and click **Create Personal Access Token**.
9. Note the token displayed in the **Your new personal access token** field (click the eye icon to see, or the copy icon to copy to the clipboard).

Troubleshooting

- If GitLab does not start, check logs:

```
sudo gitlab-ctl status  
sudo gitlab-ctl tail
```

- Restart GitLab services if needed:

```
sudo gitlab-ctl restart
```

References

- <https://docs.gitlab.com/install/azure/>
- https://livelabs.oracle.com/ords/r/dbpm/livelabs/run-workshop?p210_wid=3283&p210_wec=&session=113149767174199&P0_REDIRECT=Y

Appendix II: Setting Up NFS and Private Endpoint at Azure Cloud

Overview

Azure provides the capability to configure Network File System (NFS) file shares via **Azure NetApp Files** or **Azure Files**. This guide focuses on setting up an NFS file share using **Azure Files** with NFS protocol version 4.1, and details the steps for connecting to it.

This setup allows Linux-based workloads to mount file shares and store data in a scalable, highly available file system on Azure.

Prerequisites

Before starting, ensure you have the following:

- Before you can work with an NFS 4.1 Azure file share, you have to create an Azure storage account with the premium performance tier. Currently, NFS 4.1 shares are only available as premium file shares.
- Azure subscription (Sign in to the [Azure portal](#).)
- **Azure CLI** - Installed and configured on your local machine or VM.
- **Azure Storage Account** - To store your file shares.
- **Appropriate Permissions** - Ensure you have contributor access to the subscription and resources.

Create a File storage account

1. On the Azure portal menu, select **All services**. In the list of resources, type **Storage Accounts**. As you begin typing, the list filters based on your input. Select **Storage Accounts**.
2. On the **Storage Accounts** window that appears, choose **+ Create**.
3. On the **Basics** tab, select the subscription in which to create the storage account.
4. Under the **Resource group** field, select **Create new** to create a new resource group to use for this tutorial.
5. Enter a name for your storage account. The name you choose must be unique across Azure. The name also must be between 3 and 24 characters in length, and may include only numbers and lowercase letters.
6. Select a region for your storage account, or use the default region. Azure supports NFS file shares in all the same regions that support premium file storage.
7. Select the *Premium* performance tier to store your data on solid-state drives (SSD). Under **Premium account type**, select *File shares*.
8. Leave replication set to its default value of *Locally redundant storage (LRS)*.
9. Select **Review + Create** to review your storage account settings and create the account.
10. When you see the **Validation passed** notification appear, select **Create**. You should see a notification that deployment is in progress.

The following image shows the settings on the **Basics** tab for a new storage account:

Figure 2. Azure Screenshot on storage account

Create a storage account - Microsoft Azure

portal.azure.com/#create/Microsoft.StorageAccount

Microsoft Azure

All services > Storage accounts >

Create a storage account

Basics | Advanced | Networking | Data protection | Encryption | Tags | Review + create

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription * Visual Studio Enterprise Subscription

Resource group * (New) ExampleResourceGroup
[Create new](#)

Instance details

If you need to create a legacy storage account type, please click [here](#).

Storage account name * examplestorageaccount000

Region * (US) East US

Performance *
☐ Standard: Recommended for most scenarios (general-purpose v2 account)
☒ Premium: Recommended for scenarios that require low latency.

Premium account type * File shares

Redundancy * Locally-redundant storage (LRS)

[Review + create](#) < Previous Next: Advanced >

Deploy an Azure VM running Linux

Next, create an Azure VM running Linux to represent the on-premises server. When you create the VM, a virtual network will be created for you. The NFS protocol can only be used from a machine inside of a virtual network.

1. Select **Home**, and then select **Virtual machines** under **Azure services**.
2. Select **+ Create** and then **+ Azure virtual machine**.
3. In the **Basics** tab, under **Project details**, make sure the correct subscription and resource group are selected. Under **Instance details**, type *myVM* for the **Virtual machine name**, and select the same region as your storage account. Choose your Linux distribution for your **Image**. Leave the other defaults. The default size and pricing is only shown as an example. Size availability and pricing are dependent on your region and subscription.

Figure 3. Azure Screenshot on Project details

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ	Visual Studio Enterprise Subscription
Resource group * ⓘ	ExampleResourceGroup

[Create new](#)

Instance details

Virtual machine name * ⓘ	myVM
Region * ⓘ	(US) East US
Availability options ⓘ	No infrastructure redundancy required
Security type ⓘ	Standard
Image * ⓘ	Ubuntu Server 20.04 LTS - Gen2
	See all images Configure VM generation
Azure Spot instance ⓘ	<input type="checkbox"/>
Size * ⓘ	Standard_D2s_v3 - 2 vcpus, 8 GiB memory (\$70.08/month)
	See all sizes

- Under **Administrator account**, select **SSH public key**. Leave the rest of the defaults.

Figure 4. Azure Screenshot on Administrator account

Administrator account

Authentication type ⓘ	<input checked="" type="radio"/> SSH public key <input type="radio"/> Password
	<p>i Azure now automatically generates an SSH key pair for you and allows you to store it for future use. It is a fast, simple, and secure way to connect to your virtual machine.</p>
Username * ⓘ	azureuser
SSH public key source	Generate new key pair
Key pair name *	myVM_key

- Under **Inbound port rules > Public inbound ports**, choose **Allow selected ports** and then select **SSH (22)** and **HTTP (80)** from the drop-down.

Figure 5. Azure Screenshot on Inbound port rule

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports *



None



Allow selected ports

Select inbound ports *

HTTP (80), SSH (22)



This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

6. Select the **Review + create** button at the bottom of the page.
7. On the **Create a virtual machine** page, you can see the details about the VM you are about to create. Note the name of the virtual network. When you are ready, select **Create**.
8. When the **Generate new key pair** window opens, select **Download private key and create resource**. Your key file will be download as **myVM_key.pem**. Make sure you know where the .pem file was downloaded, because you'll need the path to it to connect to your VM.

You'll see a message that deployment is in progress. Wait a few minutes for deployment to complete.

Create an NFS Azure file share

Now you're ready to create an NFS file share and provide network-level security for your NFS traffic.

Add a file share to your storage account

1. Select **Home** and then **Storage accounts**.
2. Select the storage account you created.
3. In the service menu, under **Data storage**, select **File shares**.
4. Select **+ File Share**.
5. Name the new file share *qsfileshare* and enter "100" for the minimum **Provisioned capacity**, or provision more capacity (up to 102,400 GiB) to get more performance. Select **NFS** protocol, choose a **Root Squash** setting, and select **Create**. To learn more about root squash and its security benefits for NFS file shares, see [Configure root squash for Azure Files](#).

Figure 6. Azure Screenshot on New file share

New file share

×

Name *

qsfileshare ✓

A premium file share is billed by provisioned share size, regardless of the used capacity.
[Learn more](#)

- The minimum share size is 100 GiB.
- Provision more capacity to get more performance.

Provisioned capacity * ⓘ

100 ✓

[Set to maximum](#) GiB

Performance

Maximum IO/s ⓘ	3100
Burst IO/s ⓘ	10000
Throughput rate ⓘ	110.0 MiBytes / s

Protocol * ⓘ

☐ SMB
☒ NFS

Root Squash * ⓘ

No Root Squash ▼

⚠ Access control for NFS shares is provided by Virtual Networks. You can create this NFS file share now, but you must configure a virtual network rule before the file share can be accessed.

Create

Cancel

Set up a private endpoint or service endpoint

Next, set up a private endpoint for your storage account. This gives your storage account a private IP address from within the address space of your virtual network. Standard [data processing rates](#) for private endpoints apply. If you don't require a static IP address, you can use a service endpoint instead. There's no extra charge for using service endpoints.

1. Select the file share *qsfileshare*. You should see a dialog that says *Connect to this NFS share from Linux*. Under **Network configuration**, select **Review options**

Figure 7. Azure Screenshot on NFS Share connection

🔗 Change size and performance

🗑 Delete share

Connect from Linux

Configure network security

Connect to this NFS share from Linux

This file share supports the NFS 4.1 protocol with full POSIX semantics. You can mount this share on any OS with an NFS 4.1 client, including most Linux distributions.

⤴

✖ Network configuration

The storage account is currently not configured for use from a virtual network. The NFS protocol can only be used from a machine inside of a virtual network.


Review options

- Next, select **Setup a private endpoint**.

Figure 8. Azure Screenshot on Network security options.

Network security options

The use of the NFS protocol for an Azure file share requires network-level security configurations and disabling the "secure transfer required" storage account configuration.




Private endpoint

A private endpoint gives your storage account a private, static IP address within your virtual network, preventing connectivity interruptions from dynamic IP address changes. Traffic to your storage account stays within peered virtual networks, including those in other regions and on premises. [Learn more.](#)

Standard data processing rates for private endpoints apply. [Learn more.](#)


Setup a private endpoint



Service endpoint

If you don't require a static IP address, you can enable a service endpoint for Azure Files within the virtual network. A service endpoint configures storage accounts to allow access only from specific subnets. The allowed subnets can belong to a virtual network in the same subscription or a different subscription, including those that belong to a different Azure AD tenant. There's no extra charge for using service endpoints. [Learn more.](#)

Configure service endpoint



Connect from on-premises

You may want machines from outside of Azure to use the NFS protocol with this share. These articles describe options for connecting to the vnet you've configured for this storage account and share. A private endpoint may be required.

[Azure Express route](#)
[Site to site VPN](#)
[Point to site VPN](#)

- Select **+ Private endpoint**.

Figure 9. Azure Screenshot on Private endpoint

Firewalls and virtual networks
Private endpoint connections

+ **Private endpoint**

✓ Approve
✗ Reject
🗑 Remove
🔄 Refresh

- Leave **Subscription** and **Resource group** the same. Under **Instance**, provide a name and select a region for the new private endpoint. Your private endpoint must be in the same region as your virtual network, so use the same region as you specified when creating the VM. When all the fields are complete, select **Next: Resource**.

Figure 10. Azure Screenshot on create private endpoint

Create a private endpoint ...

1 Basics 2 Resource 3 Virtual Network 4 DNS 5 Tags 6 Review + create

Use private endpoints to privately connect to a service or resource. Your private endpoint must be in the same region as your virtual network, but can be in a different region from the private link resource that you are connecting to. [Learn more](#)

Project details

Subscription * ⓘ Visual Studio Enterprise Subscription

Resource group * ⓘ ExampleResourceGroup
[Create new](#)

Instance details

Name * mynewprivateendpoint ✓

Network Interface Name * mynewprivateendpoint-nic ✓

Region * East US

- Confirm that the **Subscription**, **Resource type** and **Resource** are correct, and select **File** from the **Target sub-resource** drop-down. Then select **Next: Virtual Network**.

Figure 10. Azure Screenshot on create private endpoint

Create a private endpoint ...

✓ Basics 2 Resource 3 Virtual Network 4 DNS 5 Tags 6 Review + create

Private Link offers options to create private endpoints for different Azure resources, like your private link service, a SQL server, or an Azure storage account. Select which resource you would like to connect to using this private endpoint. [Learn more](#)

Subscription Visual Studio Enterprise Subscription

Resource type Microsoft.Storage/storageAccounts

Resource examplestorageaccount000

Target sub-resource * ⓘ file

- Under **Networking**, select the virtual network associated with your VM and leave the default subnet. Under **Private IP configuration**, leave **Dynamically allocate IP address** selected. Select **Next: DNS**.

Figure 11. Azure Screenshot on create private endpoint

Create a private endpoint ...

✓ Basics ✓ Resource **3 Virtual Network** ④ DNS ⑤ Tags ⑥ Review + create

Networking

To deploy the private endpoint, select a virtual network subnet. [Learn more](#)

Virtual network * ⓘ

ExampleResourceGroup-vnet

Subnet * ⓘ

ExampleResourceGroup-vnet/default (10.1.0.0/24)

Network policy for private endpoints

Disabled ([edit](#))

Private IP configuration

☒ Dynamically allocate IP address

☐ Statically allocate IP address

Application security group

Configure network security as a natural extension of an application's structure. ASG allows you to group virtual machines and define network security policies based on those groups. You can specify an application security group as the source or destination in an NSG security rule. [Learn more](#)

[+](#) Create

Application security group

6. Select **Yes** for **Integrate with private DNS zone**. Make sure the correct subscription and resource group are selected, and then select **Next: Tags**.

Figure 12. Azure Screenshot on create private endpoint

Create a private endpoint ...

✓ Basics ✓ Resource ✓ Virtual Network **4 DNS** ⑤ Tags ⑥ Review + create

Private DNS integration

To connect privately with your private endpoint, you need a DNS record. We recommend that you integrate your private endpoint with a private DNS zone. You can also utilize your own DNS servers or create DNS records using the host files on your virtual machines. [Learn more](#)

Integrate with private DNS zone

☒ Yes ☐ No

Configuration name	Subscription	Resource group	Private DNS zone
privatelink-file-core-wind...	Visual Studio Enterpr... <input type="text"/>	ExampleResourceGro... <input type="text"/>	(new) privatelink.file.core....

- You can optionally apply tags to categorize your resources, such as applying the name **Environment** and the value **Test** to all testing resources. Enter name/value pairs if desired, and then select **Next: Review + create**.

Figure 13. Azure Screenshot on create private endpoint

Create a private endpoint ...

✓ Basics ✓ Resource ✓ Virtual Network ✓ DNS **5 Tags** 6 Review + create

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. [Learn more about tags](#)

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name	Value	Resource
environment	test	2 selected
		2 selected

- Azure will attempt to validate the private endpoint. When validation is complete, select **Create**. You'll see a notification that deployment is in progress. After a few minutes, you should see a notification that deployment is complete.

Disable secure transfer

Azure Files doesn't currently support encryption-in-transit with the NFS protocol and relies instead on network-level security. Therefore, you'll need to disable secure transfer.

- Select **Home** and then **Storage accounts**.
- Select the storage account you created.
- In the service menu, under **Data storage**, select **File shares**.
- Select the NFS file share that you created. Under **Secure transfer setting**, select **Change setting**.

Figure 11. Azure Screenshot on Disable secure transfer

Change size and performance Delete share

Connect from Linux Configure network security

Connect to this NFS share from Linux

This file share supports the NFS 4.1 protocol with full POSIX semantics. You can mount this share on any OS with an NFS 4.1 client, including most Linux distributions.

✓ Network configuration

^ ✗ Secure transfer setting

"Secure transfer required" is a setting that is enabled for this storage account. The NFS protocol does not support encryption and relies on network-level security. This setting must be disabled for NFS to work.

Change setting

- Change the **Secure transfer required** setting to **Disabled**, and select **Save**. The setting change can take up to 30 seconds to take effect.

Figure 12. Azure Screenshot on disable secure transfer

Save Discard Refresh

The cost of your storage account depends on the usage and the options you choose below. [Learn more about storage pricing](#)

Account kind
FileStorage

Performance ☐ Standard ☒ Premium

This setting cannot be changed after the storage account is created.

Secure transfer required ☒ Disabled ☐ Enabled

Allow Blob public access ☐ Disabled ☒ Enabled

Allow storage account key access ☐ Disabled ☒ Enabled

Allow recommended upper limit for shared access signature (SAS) expiry interval ☒ Disabled ☐ Enabled

Default to Azure Active Directory authorization in the Azure portal ☒ Disabled ☐ Enabled

Minimum TLS version

Replication

Connect to your VM

Create an SSH connection with the VM.

1. Select **Home** and then **Virtual machines**.
2. Select the Linux VM you created for this tutorial and ensure that its status is **Running**. Take note of the VM's public IP address and copy it to your clipboard.

Figure 13. Azure Screenshot on VM configuration

myVM Virtual machine

Search (Ctrl+F)

Connect Start Restart Stop Capture Delete Refresh Open in mobile CLI / PS Feedback

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Networking

Connect

Disks

Size

Security

Advisor recommendations

Extensions + applications

Continuous delivery

Availability + scaling

Configuration

Identity

Properties

Locks

Operations

Bastion

Auto-shutdown

Backup

Essentials

Resource group (move) : [ExampleResourceGroup](#)

Status : Running

Location : East US

Subscription (move) : [Visual Studio Enterprise Subscription](#)

Subscription ID : xxxxxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

Tags (edit) : [Click here to add tags](#)

Properties Monitoring Capabilities (7) Recommendations Tutorials

Virtual machine

Computer name : myVM

Health state : -

Operating system : Linux (ubuntu 18.04)

Publisher : Canonical

Offer : UbuntuServer

Plan : 18_04-lts-gen2

VM generation : V2

Agent status : Ready

Agent version : 2.7.0.6

Host group : None

Host : -

Proximity placement group : -

Colocation status : N/A

Capacity reservation group : -

Availability + scaling

Availability zone : -

Scale Set : -

Networking

Public IP address : xxxxxxxx

Public IP address (IPv6) : -

Private IP address : 10.0.0.4

Private IP address (IPv6) : -

Virtual network/subnet : [ExampleResourceGroup-vnet/default](#)

DNS name : [Configure](#)

Size

Size : Standard D4s v3

vCPUs : 4

RAM : 16 GiB

Disk

OS disk : myVM_OsDisk_1_45ac08ba8e84d00a97768c0da294407

Encryption at host : Disabled

Azure disk encryption : Not enabled

Ephemeral OS disk : N/A

Data disks : 0

3. If you are on a Mac or Linux machine, open a Bash prompt. If you are on a Windows machine, open a PowerShell prompt.
4. At your prompt, open an SSH connection to your VM. Replace xx.xx.xx.xx with the IP address of your VM, and replace the path to the .pem with the path to where the key file was downloaded.

Console

```
ssh -i .\Downloads\myVM_key.pem azureuser@xx.xx.xx.xx
```

If you encounter a warning that the authenticity of the host can't be established, type **yes** to continue connecting to the VM. Leave the ssh connection open for the next step.

Tip

You can use the SSH key you created the next time you create a VM in Azure. Just select the Use a key stored in Azure for SSH public key source the next time you create a VM. You already have the private key on your computer, so you won't need to download anything.

Mount the NFS share

Now that you've created an NFS share, you have to mount it on your Linux client. Using Azure Storage Explorer isn't supported for NFS Azure file shares, either standalone or from within the Azure portal. To view the files in the share, you must mount the share.

- Select **Home** and then **Storage accounts**.
- Select the storage account you created.
- In the service menu, under **Data storage**, select **File shares**, and then select the NFS file share you created.
- You should see **Connect to this NFS share from Linux** along with sample commands to use NFS on your Linux distribution and a mounting script that contains the required mount options. For other recommended mount options, see Mount NFS Azure file share on Linux.

Important

The provided mounting script will mount the NFS share only until the Linux machine is rebooted. To automatically mount the share every time the machine reboots, see Mount an NFS share using /etc/fstab.

Figure 14 . Azure Screenshot on NFS share

[Connect from Linux](#) [Configure network security](#)

Connect to this NFS share from Linux

This file share supports the NFS 4.1 protocol with full POSIX semantics. You can mount this share on any OS with an NFS 4.1 client, including most Linux distributions.

i Having connection issues with the commands listed here? You may need to enable your subscription to use this version of Encryption in Transit. [Learn more](#)

Select your linux distribution

Debian/Ubuntu

☒ Mount with Encryption in Transit ⓘ

Sample command to use NFS on the selected Linux distribution

```
curl -sSL -O https://packages.microsoft.com/config/${source}/etc/os-release && echo "${ID}/${VERSION_ID}"/packages-microsoft-prod.deb
sudo dpkg -i packages-microsoft-prod.deb
rm packages-microsoft-prod.deb
sudo apt-get update
sudo apt-get install aznfs
```

Mount path *

/mount/eitdemo/nfsshare

Sample command to mount this NFS share

```
sudo mkdir -p /mount/eitdemo/nfsshare
sudo mount -t aznfs eitdemo.file.core.windows.net/eitdemo/nfsshare /mount/eitdemo/nfsshare -o vers=4,minorversion=1,sec=sys,nconnect=4
```

- Select your Linux distribution.
- Using the ssh connection you created to your VM, enter the sample commands to use NFS and mount the file share.

You have now mounted your NFS share, and it's ready to store files.

Troubleshooting

1. **NFS Connection Issues:**
 - Ensure the **NFS client** is installed on your Linux machine.
 - Double-check that the storage account's **networking settings** allow your machine's IP address to access the share.
 - Verify that the **NFS version** is correctly set to 4.1 in both Azure and your Linux client.
2. **Permissions:**
 - Ensure the **NFS share permissions** are set to allow access from your client's IP address or VNet.
3. **Firewall Rules:**
 - If you're using a firewall, ensure that TCP/UDP port **2049** is open for NFS traffic.

Reference

- [Create an Azure file share with Microsoft.FileShares](#)
- [Troubleshoot NFS Azure file shares](#)

Connect with us

Call **+1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

 blogs.oracle.com

 facebook.com/oracle

 twitter.com/oracle

Copyright © 2025, Oracle and/or its affiliates. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.