



# **Deploying Single-Node Tamr Core on Azure**



<b>Introduction</b>	<b>3</b>
<b>Sizing Guidelines</b>	<b>3</b>
<b>Deployment Structure</b>	<b>3</b>
Resource Groups	3
Active Directory	3
<b>Networking</b>	<b>3</b>
Virtual Networks	4
Subnets	4
Network Security Groups	4
<b>Overview of the Tamr ADLS Gen2 Terraform Module</b>	<b>4</b>
<b>Security</b>	<b>5</b>
<b>Azure Deployment Procedures</b>	<b>6</b>
Prerequisites for Deploying Azure	6
Terraform Prerequisites	7
<b>Installation Process</b>	<b>8</b>
Step 1: Configure the ADLS Gen2 Terraform Module	8
Step 2: Apply the ADLS Gen2 Terraform Module	8
Step 3: Create Active Directory Service Account & Obtain ADLS Access Information	9
Step 4: Install the Tamr Core Software Package	10
Step 5: Start ZooKeeper	10
Step 6: Set the Tamr Core License Key	10
Step 7: Start Tamr Core and Its Dependencies	10
<b>Verifying the Deployment</b>	<b>11</b>
<b>Configuring Core Connect</b>	<b>11</b>
<b>Monitoring</b>	<b>11</b>
<b>Logging</b>	<b>12</b>
<b>Backing Up and Restoring Deployments on Azure</b>	<b>12</b>

# Introduction

This document provides an overview of Tamr Core's single-node offering on Azure, basic network requirements and security features, and deployment prerequisites and steps.

Single-node deployments on Azure include:

- Tamr Core software installed on a single VM, using the Tamr Core image available in the [Azure Marketplace](#).
- ADLS Gen2 installed and configured using the Tamr-provided [ADLS Gen2 Terraform module](#). ADLS is an HDFS-compatible remote file system in Azure.

## Sizing Guidelines

See [Azure Sizing and Limits](#) for sizing guidelines.

## Deployment Structure

### Resource Groups

Resource groups allow you to organize resources—such as virtual machines or virtual networks—into groups. Every resource must be created within a resource group. However, resources in different groups can still interact with each other.

Create resource groups to help you organize the resources used by your deployment.

### Active Directory

Tamr Core integrates with LDAP. See [LDAP Authentication and Authorization](#) to configure access to Tamr Core with Active Directory.

## Networking

The deployment of Tamr Core on Azure uses virtual networks, subnets, and network security groups.

## Virtual Networks

Virtual networks define ranges of network addresses that resources use to communicate. In single-node Azure deployments, the Tamr Core VM uses one IP address.

## Subnets

A subnet is a block of the virtual network's IP space.

To integrate the Tamr Core deployment with your existing networking infrastructure, you must provide subnets with the correct service delegations and endpoints for the services running on that subnet. The ADLS Gen2 subnet requirements are documented with the Tamr-provided [ADLS Gen2 Terraform module](#).

- [Subnet delegations](#) allow a specific service to create and modify basic network configuration rules for a subnet to help that service communicate properly over the subnet.
- [Service endpoints](#) allow you to route traffic directly from your virtual network to the specific Microsoft Azure services you have specified, such as storage accounts and databases.

## Network Security Groups

Network security groups control the ingress and egress traffic to resources on the network. Tamr Core uses network security groups to allow Tamr Core services to communicate only over the ports that are explicitly allowed by the services that require access.

# Overview of the Tamr ADLS Gen2 Terraform Module

Terraform by Hashicorp is a cloud provider-agnostic tool for infrastructure deployments. To deploy ADLS Gen2 for Tamr Core, Tamr uses the Azure provider and the preconfigured [ADLS Gen2 Terraform module](#).




The module deploys the ADLS file system for use by Tamr, and allows you to:

- Import and export data using Core Connect.
- Backup and restore the Tamr Core deployment.

The ADLS Gen2 module consists of a root module and nested submodules. As a user of the module, you can either use the entirety of the module at its root or the nested modules directly if you need more flexibility.

The module template contains a README file that describes:

- The purpose of the module.
- The list of input values that you can specify to the module, when you run it, for each of the components that the module will be deploying.
- The list of results, or outputs, that the module produces, once you use it to deploy.

 Important: This module is subject to change. For accurate information on recent versions of the ADLS Gen2 module, refer to the README file.

## Security

Tamr Core deployment on Azure provides the following security features:

- Azure services that encrypt data at rest.
- A list of allowed IP addresses and network security group (NSG) to allow or deny network traffic to your virtual machine instances.
- Integration with LDAP and SAML for user access management.
- Encrypted, secure configuration for ZooKeeper.
- External access to the Tamr Core deployment secured through Azure Firewall configuration and HTTPS, provided by the NGINX reverse proxy server.

Note: Encryption at rest: Because a standard Tamr Core deployment on cloud infrastructure uses dedicated service instances, Tamr Core depends on service-level encryption for encryption at rest. This ensures that Tamr Core's data is encrypted using keys that are



distinct from those used by other applications, while also allowing keys to be managed using the cloud provider's standard key management service.

For non-production environments configuring a firewall, NGINX, and HTTPS are strongly recommended but not required.

**⚠ Important:** If you do not configure a firewall, NGINX, and HTTPS in a non-production deployment, all users on the network will have access to the data. Use a unique password for this deployment.

**Note:** None of the Tamr Core deployed resources are required to be configured for public access for normal operation. Tamr recommends that these resources should not be made available for public access.

# Azure Deployment Procedures

## Prerequisites for Deploying Azure

- Tamr Core Software Package from the [Azure Marketplace](#).
- Tamr Core license. Contact Tamr Support at [help@tamr.com](mailto:help@tamr.com) for a Tamr license key.
- The Azure Command-Line Interface. If you do not already have the Azure CLI installed, follow the [Microsoft Azure documentation](#) to install it.
- An Azure Subscription. When you sign in to the Azure CLI for the first time, configure your account and provide your subscription as follows:  

```
az account set --subscription <my_subscription>
```
- Contributor Role. The Terraform module requires the deployment user or service principal to have a [Contributor role](#).
- The Terraform Software Package. Install Terraform on the machine on which you intend to run Terraform templates for the Tamr Core deployment. Refer to the README file for the ADLS Gen2 module, included with the module source files in the repository, for the supported Terraform version and AWS provider plugin package version. The Azure Provider plugin package is frequently upgraded.



For information on the Azure plugin releases, see [Releases for terraform-provider-azurerm](#). To ensure that Terraform uses the correct version of the provider, the provider version is included in the configuration in the `provider.tf` file. When you run `terraform init`, Terraform downloads the specified version of that provider's plugin.

-Tamr Terraform Module for ADLS Gen2. Access the Tamr Terraform module from the [GitHub repository](#).

- NGINX reverse proxy server. Configure secure external access to Tamr Core via HTTPS via a reverse proxy from the NGINX application server. For more information, see [Requirements](#) (for NGINX version support), [Installing NGINX](#), and [Configuring HTTPS](#).
- Azure Firewall. Deploy and configure the Azure Firewall using the Azure portal. See [Deploy and Configure the Azure Firewall Using the Azure Portal](#) in the Azure documentation for instructions. Firewall configuration requirements:
  - Allow only internal access to Tamr Core default port 9100 (via TCP).
  - Open port 443 for HTTPS, with a restrictive IP range that you specify using IPv4 addresses in CIDR notation, such as `1.2.3.4/32`.  
Note: If you plan to forward HTTP traffic to HTTPS, also open port 80.

## Terraform Prerequisites

- Tenant ID and Subscription ID. These values can be found with the Azure command line:  


```
az account list
```
- Resource Group for Tamr Core and its Required VMs, Networks, and Subnets. For organizational purposes, Tamr recommends creating a dedicated resource group for Tamr Core. This resource group can include all of the related resources for the deployment that you want to manage as a group. You can create this resource group with Terraform. See [azurerm\\_resource\\_group](#) in the Terraform documentation. You can also create a resource group in the Azure Portal or with the Azure CLI. See [Create Resource Groups](#) in the Microsoft Azure documentation.
- Virtual Networks and Subnets. Create a dedicated virtual network where you intend to deploy Tamr Core.



Add `Microsoft.AzureActiveDirectory` and `Microsoft.Storage` to the list of the allowed service endpoints on this subnet. This allows the ADLS service endpoint to securely connect to a Tamr Core VM on the subnet. This is required if you are loading source data from ADLS.

## Installation Process

### Step 1: Configure the ADLS Gen2 Terraform Module

 Important: Before you begin this task, verify that you have configured the required virtual networks and subnets as specified above. Tamr also recommends creating a resource group in which to provision the Tamr Core resources instead of using an existing resource group.

The source files for the ADLS Gen2 module in the [repository](#) include:

- The `README.md` file that describes how to use the module.
- Example uses of the modules in the `/examples` folder.

Fill in the required parameters and any other optional parameters you may require for your deployment.

For more detail, see the example module uses in each module's `/examples` folder.

Note: Tamr's Terraform modules follow their own release cycles. To use a newer version, update the version in the `ref` query parameter in the source variable as shown below:

Update Terraform Version

```
module "example_source_reference" {  
  source = "git::https://github.com/Datatamer/<name of repo>?ref=0.1.0"  
}
```

### Step 2: Apply the ADLS Gen2 Terraform Module

Tip: There are many possible workflows for applying Terraform modules. If you are proficient in Terraform, feel free to adapt this workflow to suit your situation.





To apply the ADLS Gen2 Terraform modules:

1. On the machine from which you intend to control the deployments, sign in to Azure:  
`az login`
2. Initialize the modules in the directory with your Terraform code:  
`terraform init`
3. Review the resources that are being provisioned, changed, or deleted:  
`terraform plan`
4. Apply the plan:  
`terraform apply`

Note: Some deprecation warnings may appear while the Terraform commands run. You can ignore these messages.

## Step 3: Create Active Directory Service Account & Obtain ADLS Access Information

Create an Azure AD application and service principal that can access resources as described in the [Microsoft documentation](#).

Add the following permissions to the application:

- Azure Storage user\_impersonation  
`https://storage.azure.com/user_impersonation`
- Azure Data Lake user\_impersonation  
`https://datalake.azure.net/user_impersonation`  
See the [Microsoft documentation](#) for details.

Assign the following roles:

- Storage Account Contributor
- Storage Blob Data Contributor  
See the [Microsoft documentation](#) for details.



Once created, retrieve the following information:

1. Record the Application (client) ID and tenant ID from the portal as described in the [Microsoft documentation](#).
2. On the left-hand side, select Certificates & secrets.
3. Add a new client secret key by selecting New client secret as described in the [Microsoft documentation](#).
4. Record the value of your new client secret.

## Step 4: Install the Tamr Core Software Package

Obtain the Tamr Core image from the [Azure Marketplace](#).

To install Tamr Core, SSH into the Tamr Core VM and follow the procedure for [installing](#) Tamr Core on the VM.

Note that as part of the Azure marketplace image:

- The software bundle is not packaged in a zip file.
- PostgreSQL is automatically installed and configured on the Tamr Core VM.

## Step 5: Start ZooKeeper

Run `<tamr-home-dir>/tamr/start-zk.sh` to start ZooKeeper.

## Step 6: Set the Tamr Core License Key

Set the license key for Tamr Core by following the [procedure to create and upload a yaml configuration file](#).

## Step 7: Start Tamr Core and Its Dependencies

For detailed instructions, follow the [procedure for installing Tamr Core](#).



In general, run these commands:

Commands to start Tamr

```
<tamr-home-dir>/start-dependencies.sh
```

```
<tamr-home-dir>/start-unify.sh
```

## Verifying the Deployment

To verify that HBase and Spark are functioning properly:

1. SSH to the VM on which Tamr Core is installed.
2. Navigate to `http://<tamr-vm-ip>:9100` and sign in.
3. Upload a small CSV dataset and profile it.

To verify that Elasticsearch is functioning properly:

1. Create a schema mapping project.
2. Add the dataset you just uploaded to the project.
3. Bootstrap several attributes.
4. Run Update Unified Dataset.
5. Verify that you now see records on the Unified Dataset page.

For more detailed instructions, see [Tamr Core Installation Verification Steps](#).

## Configuring Core Connect

To move data files from cloud storage into Tamr Core, and exported datasets from Tamr Core to cloud storage, you use the Core Connect service. See [Configuring Core Connect](#).

## Monitoring

You can use Azure Monitor as your cloud monitoring tool, allowing you to monitor and analyze applications. The tool offers observability into infrastructure and network performance in real time.

## Logging

Use the logs provided by Tamr Core for single-node deployments. See [Logging in Single-Node Deployments](#).

## Backing Up and Restoring Deployments on Azure

See [Configuring Tamr Core Backup](#) for backup configuration instructions.

To backup and restore deployments on Azure, see [Backup](#) and [Restore](#).