opentext™

OpenText™ eDirectory CE 24.4 (v9.3)

Installation Guide

January 2025

Legal Notice

Copyright 2009 - 2024 Open Text All rights reserved.

The only warranties for products and services of Open Text and its affiliates and licensors (“Open Text”) are as may be set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Open Text shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Contents

About this Book and the Library 9

1 Migrating OpenText eDirectory to 24.4 (v9.3) 11

Prerequisites for Migrating OpenText eDirectory . . . 11
Important Changes in OpenText eDirectory Following OpenSSL 3.0 Adoption . . . 12
Migrating OpenText eDirectory to 9.3 . . . 14

2 Install and Upgrade Features 15

Multiple Package Formats for Installing OpenText eDirectory 9.3 . . . 16
Installing OpenText eDirectory 9.3 in a Custom Location . . . 16
Specifying a Custom Location for Application Files . . . 16
Specifying a Custom Location for Data Files . . . 17
Specifying a Custom Location for Configuration Files . . . 17
Non-root Install . . . 18
Standards Compliance . . . 18
FHS Compliance . . . 18
LSB Compliance . . . 19
Server Health Checks . . . 20
Need for Health Checks . . . 20
What Makes a Server Healthy? . . . 20
Performing Health Checks . . . 20
Types of Health Checks . . . 21
Categorization of Health . . . 22
Log Files . . . 23
Installing Bridge Patch to the EBA Enabled Server . . . 23
Installing Bridge Patch to Multiple Replica Servers with EBA in Linux . . . 24
Installing Bridge Patch to Multiple Replica Servers with EBA in Windows . . . 25
Installing Bridge Patch to Multiple Replica Servers with EBA in Container . . . 26
For More Information . . . 26

3 Installing or Upgrading OpenText eDirectory CE 24.4 (v9.3) on Linux 27

System Requirements . . . 27
Prerequisites . . . 29
Hardware Requirements . . . 33
Forcing the Backlink Process to Run . . . 33
Adding the edirAdmin User Group . . . 33
Upgrading OpenText eDirectory . . . 34
Server Health Checks . . . 35
Upgrading on Linux Servers Other Than OES . . . 35
Unattended Upgrade of OpenText eDirectory on Linux . . . 36
Upgrading the Tarball Deployment of eDirectory 9.3 . . . 37
Non-Root Users Upgrading OpenText eDirectory 9.3 . . . 38
Upgrading Multiple Instances . . . 39
Installing OpenText eDirectory . . . 40

Using SLP with OpenText eDirectory 40
Using the nds-install Utility to Install OpenText eDirectory Components 41
Non-root User Installing OpenText eDirectory 9.3 44
Non-Root User Configuring OpenText eDirectory with Root Install 48
Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server 49
Using ndsconfig to Configure Multiple Instances of OpenText eDirectory 9.3 56
Using ndsconfig to Install a Linux Server into a Tree with Dotted Name Containers 63
Using the nmasinst Utility to Configure NMAS 63
Non-root User SNMP Configuration 64
Locating Log Files 65

4 Installing or Upgrading OpenText eDirectory CE 24.4 (v9.3) on Windows 67

System Requirements 67
Prerequisites 68
Hardware Requirements 70
Forcing the Backlink Process to Run 71
Installing OpenText eDirectory on Windows 71
Installing or Updating OpenText eDirectory 9.3 on a Windows Server 72
Server Health Checks 73
Communicating with OpenText eDirectory through LDAP 74
Installing NMAS Server Software 75
Installing into a Tree with Dotted Name Containers 75
Unattended Install and Configure to OpenText eDirectory 9.3 on Windows 76
Locating Log Files 82
Upgrading OpenText eDirectory on Windows 83
NMAS Schema Extension before Upgrading OpenText eDirectory 83
Upgrading OpenText eDirectory Using Windows Installer 84
Unattended Upgrade of OpenText eDirectory on Windows 85

5 Deploying OpenText eDirectory on Microsoft Azure 87

Prerequisites 87
Deployment Procedure 87
Deploying on a Linux Platform 87
Deploying on a Windows Platform 101
Deploying OpenText eDirectory Container on Microsoft Azure Container Instance 108
Checklist for Deploying the Container 108

6 Deploying OpenText eDirectory on Amazon Web Services EC2 115

Prerequisites 115
Deployment Procedure 115
Preparing AWS Virtual Private Cloud 117
Configuring Network ACLs 118
Configuring Security Groups 120
Creating a SSH Key Pair 121
Creating and Deploying Instances 121
Configuring EBS Volume for Storing OpenText eDirectory Data 121
Installing OpenText eDirectory and OpenText Identity Console 122
Deploying Auditing Services 126
Disaster Recovery 126
Sizing Guidance for OpenText eDirectory Deployment on AWS 127

Get Started with the AWS EC2 Instance Selection 128
Details of Performance Test Environments 128
Determining the Sizing Requirement 130

7 Deploying OpenText eDirectory Using Docker Container 135

Why Docker? 135
Planning to Deploy OpenText eDirectory Using Docker Container 135
System Requirements 135
Prerequisites 136
Docker CLI 136
Deploying OpenText eDirectory Container 136
Deploying OpenText eDirectory Container in Host Network 138
Deploying OpenText eDirectory Container in User Defined Overlay Network 140
Post-Deployment Tasks 142
Executing Commands on a Running OpenText eDirectory Container 142
Configuring OpenSLP for OpenText eDirectory Docker Container 143
Installing NMAS Methods in OpenText eDirectory Docker Container 143
Installing New Packages in OpenText eDirectory Docker Container 144
Configuring the CEF Property File in OpenText eDirectory Docker Container 144
Managing OpenText eDirectory Data Storage 145
Upgrading OpenText eDirectory Using Docker Container 145
Recovering OpenText eDirectory Docker Containers 146

8 Installing OpenText eDirectory on Linux and Windows with IPv6 Addresses 147

Configuring OpenText eDirectory on Linux with IPv6 148
Creating a New OpenText eDirectory Tree 148
Adding a Server to an Existing OpenText eDirectory Tree 148
Enabling IPv6 Addresses on Existing or Upgraded OpenText eDirectory Servers 148
Adding LDAP URLS for IPV6 on the LDAP Server Object 149
Installing or Upgrading OpenText eDirectory on Windows with IPv6 149
Enabling IPv6 While Installing or Upgrading OpenText eDirectory 149
Enabling IPv6 for Existing Servers 149
Accessing OpenText iMonitor 150

9 Operating OpenText eDirectory in FIPS Mode 151

Configuring OpenText eDirectory in FIPS Mode 151

10 Relocating the DIB 153

Linux 153
Windows 154

11 Upgrade Requirements of OpenText eDirectory 9.3 155

Reference Changes in 8.8 SP1 or Later Versions 155
Upgrade Process in 9.3 156

12 Configuring OpenText eDirectory on Linux 157

Configuration Utilities 157

The ndsconfig Utility 157
Using LDAP Tools to Configure the LDAP Server and LDAP Group Objects 158
Using the nmasinst Utility to Configure NetIQ Modular Authentication Service 158
Customizing OpenText eDirectory 158
Configuration Parameters 159
Security Considerations 165

13 Deploying OpenText eDirectory on High Availability Clusters 167

Clustering OpenText eDirectory Services on Linux 168
Prerequisites 168
Installing and Configuring OpenText eDirectory 168
Configuring SNMP Server in Clustered Linux Environments 170
Clustering OpenText eDirectory Services on Windows 171
Prerequisites 171
Installing and Configuring OpenText eDirectory 171
Configuring SNMP Server in Clustered Windows Environments 173
Troubleshooting Clustered Environments 173
Repairing or Upgrading OpenText eDirectory on Clustered Nodes 173
Creating Windows Registry Keys 173
Configuration Utility Options 174

14 Uninstalling OpenText eDirectory 175

Uninstalling OpenText OpenText eDirectory on Windows 175
Uninstalling OpenText eDirectory, ConsoleOne, and SLP DA 175
Unattended Uninstallation of OpenText eDirectory 176
Uninstalling NICI 179
Uninstalling Microsoft Visual C++ 2005 and Visual C++ 2012 Runtime Libraries 179
Uninstalling OpenText eDirectory on Linux 180
Unattended Uninstallation of OpenText eDirectory on Linux 181
Caveats for Uninstalling OpenText eDirectory 181

A Linux Packages for OpenText eDirectory 183

B OpenText eDirectory Health Checks 187

Need for Health Checks 187
Performing Health Checks 187
With the Upgrade 187
As a Standalone Utility 188
Types of Health Checks 188
Basic Server Health 188
Partitions and Replica Health 189
Categorization of Health 189
Normal 189
Warning 189
Critical 190
Log Files 190

C Configuring OpenSLP for OpenText eDirectory 193

Service Location Protocol 193

SLP Fundamentals 193
OpenText Service Location Providers 194
User Agents 194
Service Agents 195
Configuration Parameters 195

D Troubleshooting Issues 197

Troubleshooting the Installation Issues 197
Troubleshooting the Configuration Issues 199
Troubleshooting the Issues with Multiple Instances of OpenText eDirectory 201
ndsconfig Utility 202
Troubleshooting NMAS Installation 202
Troubleshooting Certificate Server Installation 203
Troubleshooting Installation 204

About this Book and the Library

The *Installation Guide* describes how to install OpenText™ eDirectory 9.3. It is intended for network administrators.

For the most recent version of the *OpenText eDirectory CE 24.4 (v9.3) Installation Guide*, see the OpenText™ eDirectory online documentation (https://www.netiq.com/documentation/edirectory-93/) Website.

Intended Audience

The guide is intended for network administrators.

Other Information in the Library

The library provides the following information resources:

Administration Guide

Describes how to manage and configure OpenText eDirectory.

Tuning Guide for Linux Platforms

Describes how to analyze and tune OpenText eDirectory on Linux platforms to yield superior performance in all deployments.

These guides are available at the OpenText™ eDirectory CE 24.4 (v9.3) (https://www.netiq.com/documentation/edirectory-93/).

For information about the OpenText eDirectory management utility, see the OpenText™ Identity Console Administration Guide.

1 Migrating OpenText eDirectory to 24.4 (v9.3)

OpenText eDirectory 9.3 supports OpenSSL 3.0. You must perform the migration steps to move your existing OpenText eDirectory setup to version 9.3. This section outlines the steps to migrate OpenText eDirectory from 9.2 to 9.3.

- "Prerequisites for Migrating OpenText eDirectory" on page 11
- "Important Changes in OpenText eDirectory Following OpenSSL 3.0 Adoption" on page 12
- "Migrating OpenText eDirectory to 9.3" on page 14

Prerequisites for Migrating OpenText eDirectory

Before beginning the migration, ensure to complete the following:

- ❒ **Backup:** Perform a complete backup of your OpenText eDirectory instance, including configuration files and data.
- ❒ **System Compatibility:** OpenSSL 3.0 is only supported on SLES 15 SP3, SP4, SP5 and SP6 and RHEL 8.8, 8.9 and 8.10 and RHEL 9.2, 9.3 and 9.4. Ensure your platform meets this requirement.
- ❒ **Patching:** Apply the latest patches to your existing OpenText eDirectory installation to version 9.2 or later.
- ❒ **Bridge Patch:** If you have a multi-server environment and if EBA is enabled in any or all of the servers, then before upgrading to OpenText eDirectory 9.3.0, **it is mandatory to upgrade all OpenText eDirectory Servers to latest OpenText eDirectory 9.2.8 or 9.2.9 versions.** Post Successful Upgrade, If you have multi server setup, user must apply the EBA bridge patch to all the servers, master and replica where EBA is enabled. EBA bridge patch is available with OpenText eDirectory build in the following paths -
  - **For Linux:**

    `<Untarred_edirectory_location>/eDirectory/eba_bridge_patch`
  - **For Windows:**

    `<extracted_patch>eDirectory\eba_bridge_patch`
  - **For Container:**

    `<Untarred_edirectory_location>/eDirectory_930/eba_bridge_patch`

  Execute the following command to check if EBA is enabled in the system.

  ```
  ndscheck -a  <admin_credentials> -w <tree_login_password>
  ```

  For more details, see "Installing Bridge Patch to Multiple Replica Servers with EBA in Linux" on page 24, "Installing Bridge Patch to Multiple Replica Servers with EBA in Windows" on page 25 and "Installing Bridge Patch to Multiple Replica Servers with EBA in Container" on page 26.

- **Same Box compatibility:** If an older version of Identity Console is installed on the same system as OpenText eDirectory, upgrade the Identity Console to version 1.9 or later. During the upgrade, the OpenText eDirectory installation will exit if it detects an older version of Identity Console on the same machine.
- **Support for 56-Bit/ DES Key:** From eDirectory 24.4(v9.3), 56-bit/ DES Key configurations are no longer supported, with or without FIPS mode. This is due to OpenSSL 3.0, which supports stronger key configurations. For guidance on identifying and migrating away from 56-bit keys, refer to the following TID: KM000040457 (https://portal.microfocus.com/s/article/KM000040457?language=en_US).

NOTE: ◆During installation, OpenText eDirectory installer checks for the bridge patch applied in all the replica server with EBA and exits if its not applied to any of the replica. EBA check does not work on non-root user and container.

- From OpenText eDirectory 9.3.0 onwards, if it is 9.3.0 master server then adding any lower versions of OpenText edirectory as replica is not supported, with or without EBA.
- If OpenText eDirectory 9.2.9 or 9.2.8 coexists with the OpenText Identity Console 1.8, then it is recommended to upgrade OpenText Identity Console to 1.9 before upgrading OpenText eDirectory to 9.3 on Linux.

Important Changes in OpenText eDirectory Following OpenSSL 3.0 Adoption

The following are the important changes you must understand before starting the migration.

TLS Protocols and Cipher Suites

- The product now supports TLS 1.3 alongside TLS 1.2.
- Cipher suites have been updated to align with OpenSSL 3.0 and include the following TLS 1.3 Ciphers suites:
  - TLS_AES_256_GCM_SHA384.
  - TLS_CHACHA20_POLY1305_SHA256 (non-FIPS only).
  - TLS_AES_128_GCM_SHA256.
- TLS 1.2 Ciphers continue with a mix of AES and RSA encryption options.

IMPORTANT: Configure the environments requiring TLS 1.3 accordingly. Ensure to remove or update any deprecated or outdated cipher suites.

FIPS and NICI Settings

- In OpenText eDirectory 9.3, the FIPS mode is disabled by default. However, you can enable it according to the NICI (Novell International Cryptographic Infrastructure) configuration. This is applicable for TLS FIPS and NICI FIPS. In the earlier version of the product, TLS FIPS is enabled by default.
- The NICI configuration file, `nici64.cfg` becomes the reference point for enabling TLS FIPS and NICI FIPS. For more information see, Operating eDirectory in FIPS Mode (https://www.netiq.com/documentation/edirectory-93/edir_install/data/b1gnsxb4.html).

IMPORTANT: For systems requiring FIPS compliance, review the NICI configurations. If upgrading from 9.2, expect different default behaviors in TLS-FIPS settings.

RSA Certificate and Encryption Algorithms

OpenText eDirectory 9.3 no longer supports the following:

- RSA certificates with key sizes of 512, 768, or 1024 bits in the FIPS mode.
- SHA-1 certificates in the FIPS mode in OpenText eDirectory 9.3. However, these are supported in the non-FIPS configurations.
- DES and RC2-40 in the FIPS or non-FIPS environments. However, 3DES remains supported in the FIPS mode.

IMPORTANT: Review and update certificates and encryption configurations, especially if small key sizes or SHA-1 certificates are used.

NICI Tree Key Behaviour

A fresh installation of OpenText eDirectory 9.3 automatically creates an AES 256-bit tree key (CN=W1.CN=KAP.CN=Security). However, when upgrading from older OpenText eDirectory versions, the existing tree key remains unchanged and continue to function as expected. For more details about NICI keys, see Understanding the NICI Keys (https://www.netiq.com/documentation/edirectory-9/nici_admin_guide/data/agtbw4b.html)

NOTE: If you are planning on upgrading the eDirectory to 24.4(v9.3) and if **56 bit / DES tree keys** exist in the tree, then for guidance on identifying and migrating away from 56-bit keys, refer to the following TID: KM000040457 (https://portal.microfocus.com/s/article/KM000040457?language=en_US).

Backward Compatibility and Integrations

OpenText eDirectory 9.3 supports only the following versions of these products. Upgrade if you are using older versions.

- OpenText Identity Console 24.4 (v1.9)
- OpenText Identity Manager 24.4 (v4.10)

IMPORTANT: Verify the integrations with these products for version compatibility before upgrading. Older versions might cause installation failures or communication issues.

Migrating OpenText eDirectory to 9.3

Upgrade the associated components for migrating the product to 9.3.

1 OpenText Identity Console Compatibility:

OpenText eDirectory 9.3 supports only the following versions of OpenText Identity Console and OpenText Identity Manager:

- OpenText Identity Console 24.4 (v1.9)
- OpenText Identity Manager 24.4 (v4.10)

If any unsupported versions of these products are installed, upgrade these to the supported versions.

2 Handling NMAS Methods Post Upgrade

OpenText eDirectory 9.3.0 integrates NMAS Methods with the latest NICI built on OpenSSL 3.0.15.

- **Single-Server Environment:**

  No actions needed: If you have a single instance of OpenText eDirectory 9.3.0 with NMAS Methods, no additional steps are required in Linux. For windows, it is required to extend the NMAS schema. For more information, see "NMAS Schema Extension before Upgrading OpenText eDirectory" on page 83.

- **Multi-Server Environment:** If you have a multi-server setup servers with mixed versions then adding new NMAS method to OpenText eDirectory 9.3.0 does not get impacted. But, if you are in 9.2.9 or below versions of OpenText eDirectory as replica, then add the NMAS method in OpenText eDirectory 9.2.9 and then 9.3.0.

NOTE:

- **No Manual steps needed:** Once all servers are upgraded to OpenText eDirectory 9.3.0, adding NMAS Methods manually will no longer be necessary.
- For more details about NMAS Methods, see OpenText™ eDirectory CE 24.4 (v9.3) eDirectory Administration Guide (https://www.netiq.com/documentation/edirectory-9/edir_admin/data/a53vifw.html).

2 Install and Upgrade Features

This chapter discusses the features of OpenText eDirectory CE 24.4 (v9.3) installation and upgrade.

The following table lists the new features and specifies the platforms they are supported on.

Feature	Linux	Windows
Multiple package formats for installing OpenText eDirectory 9.3	✓	✗
Custom location install for application files	✓	✓
Custom location install for data files	✓	✓
Custom location install for configuration files	✓	✗
Non-root install	✓	✗
Improved support for installations on high availability clusters	✓	✓
FHS compliance	✓	✗
LSB compliance	✓	✗
Server health checks	✓	✓
SecretStore integration	✓	✓
OpenText eDirectory Instrumentation Installation	✓	✓

NOTE: ◆From OpenText eDirectory CE 24.4 (v9.3)9.2.3 onwards, AES 256-Bit Tree Key is enabled by default. But if we are upgrading from 9.2.x then one need to add W1.KAP.Security object manually after upgrading the OpenText eDirecotry.

- In EBA enabled setup, ensure that all replica servers are on OpenText eDirectory 9.2.8 or 9.2.9 if you are upgrading any of your servers to OpenText eDirectory CE 24.4 (v9.3) with EBA. Before upgrading replica to 9.3.0 apply the bridge patch. Bridge patch can not be applied on OpenText eDirectory 9.2.7 and lower versions.

This chapter includes the following information:

- “Multiple Package Formats for Installing OpenText eDirectory 9.3” on page 16
- “Installing OpenText eDirectory 9.3 in a Custom Location” on page 16
- “Non-root Install” on page 18
- “Standards Compliance” on page 18

- “Server Health Checks” on page 20
- “Installing Bridge Patch to the EBA Enabled Server” on page 23
- “For More Information” on page 26

Multiple Package Formats for Installing OpenText eDirectory 9.3

On Linux, you have an option to choose from various file formats while installing OpenText eDirectory 9.3 on your host. The file formats are listed in the table below.

Type of User and Installation Location	Linux
Root user	
Default location	RPM
Custom location	Tarball
Non-root user	
Custom location	Tarball

For more information on installing using tarballs, refer to “Upgrading the Tarball Deployment of eDirectory 9.3” on page 37.

Installing OpenText eDirectory 9.3 in a Custom Location

OpenText eDirectory 9.3 gives you the flexibility to install the application, data, and configuration files in a location of your choice.

One of the scenarios for installing OpenText eDirectory 9.3 in a custom location is when you already have an earlier version of OpenText eDirectory installed on your host and you want to test OpenText eDirectory 9.3 before upgrading to it. This way, you can have your existing OpenText eDirectory setup undisturbed and also test this new version. You can then decide whether you want to retain your existing version or want to upgrade to OpenText eDirectory 9.3.

NOTE: SLP and the SNMP sub-agent are installed in the default locations.

This section explains how to install the various files in a custom location:

- “Specifying a Custom Location for Application Files” on page 16
- “Specifying a Custom Location for Data Files” on page 17
- “Specifying a Custom Location for Configuration Files” on page 17

Specifying a Custom Location for Application Files

While installing OpenText eDirectory, you can install your application files in a location of your choice.

Linux

To install OpenText eDirectory 9.3 in a custom location, you can use the tarball installation file and untar OpenText eDirectory 9.3 in a location of your choice.

Windows

You were able to specify a custom location for the application files during the installation Wizard even prior to OpenText eDirectory 9.3.

Specifying a Custom Location for Data Files

While configuring OpenText eDirectory, you can save the data files in a location of your choice. The data files include the `data`, `dib`, and `log` directories.

Linux

To configure the data files in a custom location, you can use either the `-d` or `-D` option of the ndsconfig utility.

Option	Description
-d *custom_location*	Creates the `DIB` (the OpenText eDirectory database) directory in the path mentioned.
-D *custom_location*	Creates the `data` (contains data such as the PIDs and socket IDs), `dib`, and `log` directories in the path mentioned.

Windows

On Windows you would be prompted to enter the DIB path during the installation. Enter a path of your choice.

Specifying a Custom Location for Configuration Files

While configuring OpenText eDirectory, you can select the path where you want to save your configuration files.

Linux

To configure the `nds.conf` configuration file to a different location, use the `--config-file` option of the `ndsconfig` utility.

To install the other configuration files (such as `modules.conf`, `ndsimon.conf`, and `ice.conf`) to a different location, do the following:

1. Copy all the configuration files to the new location.
2. Set the new location by entering the following:

   ```
   ndsconfig set n4u.nds.configdir custom_location
   ```

Windows

You cannot specify a custom location for the configuration files on Windows.

Non-root Install

OpenText eDirectory 9.1 and higher supports installation and configuration of OpenText eDirectory servers by a non-root user. Earlier versions of OpenText eDirectory could be installed and configured only by a root user with only a single instance of OpenText eDirectory running on a host.

With OpenText eDirectory 9.3 or higher, any non-root user can use a tarball build to install OpenText eDirectory. There can be multiple instances of OpenText eDirectory binary installs by the same or different users. However, even for non-root user installs, the system-level services such as the Novell International Cryptographic Infrastructure (NICI), SNMP and SLP can be installed only with the root privileges. NICI is a mandatory component, and SNMP and SLP are optional components for OpenText eDirectory functionality. Also, with a package install, only a single instance can be installed by the root user.

After the install, a non-root user can configure OpenText eDirectory server instances using his or her individual tarball installation, or by using a binary installation. This means that there can be multiple instances of OpenText eDirectory servers running on a single host because any user, either root or non-root, can configure different OpenText eDirectory server instances on a single host by using either a package or tarball installation. For more details on the Multiple Instances feature, see "Upgrading Multiple Instances" on page 39.

Non-root installation and configuration is applicable to Linux platforms only. For more information on non-root installation and configuration, see "Non-root User Installing OpenText eDirectory 9.3" on page 44.

Standards Compliance

OpenText eDirectory 9.3 is compliant with the following standards:

- "FHS Compliance" on page 18
- "LSB Compliance" on page 19

FHS Compliance

To avoid file conflicts with other product application files, OpenText eDirectory 9.3 follows the File System Hierarchy Standard (FHS). This feature is available only on Linux.

OpenText eDirectory follows this directory structure only if you have chosen to install it in the default location. If you have chosen a custom location, the directory structure would be *custom_location*/*default_path*.

For example, if you choose to install in the `eDir93` directory, the same directory structure would be followed in the `eDir93` directory, like the man pages would be installed in the `/eDir93/opt/novell/man` directory.

The following table lists the change in the directory structure:

Types of Files Stored in the Directory	Directory Name and Path
Executable binaries and static shell scripts	`/opt/novell/eDirectory/bin`
Executable binaries for root use	`/opt/novell/eDirectory/sbin`
Static or dynamic library binaries	`/opt/novell/eDirectory/lib`
Configuration files	`/etc/opt/novell/eDirectory/conf`
Read/Write, run-time dynamic data like the DIB	`/var/opt/novell/eDirectory/data`
Log files	`/var/opt/novell/eDirectory/log`
Linux man pages	`/opt/novell/man`

Export Environmental Variables

With the FHS implementation in OpenText eDirectory 9.3, you need to update the path environmental variables and export them. This creates the following problems:

- You need to remember all the paths exported, so that whenever you open a shell, you need to export these paths and start using the utilities.
- When you want to use more than one set of binary, you have to open more than one shell or have to unset and set the paths to the different set of binaries frequently.

To resolve the above issue, you can use the `/opt/novell/eDirectory/bin/ndspath` script as follows:

- Prefix the ndspath script to the utility and run the utility you want as follows:

  ```
  custom_location/opt/novell/eDirectory/bin/ndspath
  utility_name_with_parameters
  ```

- Export the paths in the current shell as follows:

  ```
  . custom_location/opt/novell/eDirectory/bin/ndspath
  ```

- After entering the above command, run the utilities as you would normally do. Call the script in your profile, bashrc, or similar scripts. Therefore, whenever you log in or open a new shell, you can start using the utilities directly.

LSB Compliance

OpenText eDirectory 9.3 is now Linux Standard Base (LSB) compliant. LSB also recommends FHS compliance. All the OpenText eDirectory packages in Linux are prefixed with *novell*. For example, `NDSserv` is now `novell-NDSserv`.

Server Health Checks

OpenText eDirectory CE 24.4 (v9.3) includes server health checks that help you determine whether your server health is safe before upgrading.

The server health checks run by default with every upgrade and occur before the actual package upgrade. However, you can also run the diagnostic tool `ndscheck` to do the health checks.

For more information about performing routine health check procedures, see Maintaining OpenText eDirectory in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Need for Health Checks

In earlier releases of OpenText eDirectory, the upgrade did not check the health of the server before proceeding with the upgrade. If the health was unstable, the upgrade operation would fail and OpenText eDirectory would be in an inconsistent state. In some cases, you probably could not roll back to the pre-upgrade settings.

This new health check tool resolves this, letting you to ensure that your server is ready to upgrade.

What Makes a Server Healthy?

The server health check utility performs certain health checks to ensure that the tree is healthy. The tree is declared healthy when all these health checks are completed successfully.

Performing Health Checks

You can perform server health checks in two ways:

- “With the Upgrade” on page 20
- “As a Standalone Utility” on page 21

NOTE: You need administrative rights to run the health check utility. The minimal right that can be set to run the utility is the Public right. However, with the Public right some of the NetWare Core Protocol (NCP) objects and partition information are not available.

With the Upgrade

The health checks are run by default every time you upgrade OpenText eDirectory.

Linux

Every time you upgrade, the health checks are run by default before the actual upgrade operation starts.

To skip the default health checks, you can use the -j option with the `nds-install` utility.

Windows

The server health checks happen as part of the installation wizard. You can enable or disable the health checks when prompted to do so.

As a Standalone Utility

You can run the server health checks as a standalone utility any time you want. The following table explains the health check utilities.

Table 2-1 *Health Check Utilities*

Platform	Utility Name
Linux	ndscheck Syntax: `ndscheck -h hostname:port -a admin_FDN -F logfile_path --config-file configuration_file_name_and_path` **NOTE:** You can specify either `-h` or `--config-file`, but not both options.
Windows	ndscheck

Types of Health Checks

When you upgrade or run the ndscheck utility, the following types of health checks are done:

- Basic Server Health
- Partitions and Replica Health

If you run the ndscheck utility, the results from the health checks are displayed on the screen and logged in to `ndscheck.log`. For more information on log files, refer to “Log Files” on page 23.

If the health checks are done as part of the upgrade, then after the health checks, based on the criticality of the error, either you are prompted to continue the upgrade process or the process is aborted. The details of the errors are described in “Categorization of Health” on page 22.

Basic Server Health

This is the first stage of the health check.The health check utility checks for the following:

1. The OpenText eDirectory service is up. The DIB is open and able to read some basic tree information such as the tree name.
2. The server is listening on the respective port numbers.

   For LDAP, it gets the TCP and the SSL port numbers and checks if the server is listening on these ports.

   Similarly, it gets the HTTP and HTTP secure port numbers and checks if the server is listening on these ports.

Partitions and Replica Health

After checking the basic server health, the next step is to check the partitions and replica health as follows:

1. Checks the health of the replicas of the locally held partitions.
2. Reads the replica ring of each and every partition held by the server and checks whether all servers in the replica ring are up and all the replicas are in the ON state.
3. Checks the time synchronization of all the servers in the replica ring. This shows the time difference between the servers.

Categorization of Health

Based on the errors found while checking the health of a server, there can be the three categories of health. The status of the health checks is logged in to a log file. For more information, refer to "Log Files" on page 23.

The three categories of health Normal, Warning, and Critical.

Normal

The server health is normal when all the health checks were successful.

The upgrade proceeds without interruption.

Warning

The server health is in the warning category when minor errors are found while checking the health.

If the health check is run as part of the upgrade, you are prompted to either abort or continue.

Warnings normally occur in the following scenarios:

1. Server not listening on LDAP and HTTP ports, either normal or secure or both.
2. Unable to contact any of the nonmaster servers in the replica ring.
3. Servers in the replica ring are not in sync.

Critical

The server health is critical when critical errors were found while checking the health.

If the health check is run as part of the upgrade, the upgrade operation is aborted.

The critical state normally occurs in the following cases:

1. Unable to read or open the DIB. The DIB might be locked or corrupt.
2. Unable to contact all the servers in the replica ring.
3. Locally held partitions are busy.
4. Replica is not in the ON state.

Log Files

Every server health check operation, whether it is run with the upgrade or as a standalone utility, maintains the status of the health in a log file.

The content of the log file is similar to the messages displayed on the screen when the checks are happening.

The health check log file contains the following:

- Status of the health checks (normal, warning, or critical).
- URLs to the OpenText support site.

The following table gives you the locations for the log file on the various platforms:

Table 2-2 *Health Check Log File Locations*

Platform	Log File Name	Log File Location
Linux	`ndscheck.log`	Depends on the location you specified with the `ndscheck -F` utility. If you did not use the `-F` option, the location of the `ndscheck.log` file is determined by the other options you used at the ndscheck command line as follows: 1. If you used the `-h` option, the `ndscheck.log` file is saved in the user's home directory. 2. If you used the `--config-file` option, the `ndscheck.log` file is saved in the server instance's log directory. You can also select an instance from the multiple instances list.
Windows	`ndscheck.log`	*`install_directory`*

Installing Bridge Patch to the EBA Enabled Server

- "Installing Bridge Patch to Multiple Replica Servers with EBA in Linux" on page 24
- "Installing Bridge Patch to Multiple Replica Servers with EBA in Windows" on page 25
- "Installing Bridge Patch to Multiple Replica Servers with EBA in Container" on page 26

Installing Bridge Patch to Multiple Replica Servers with EBA in Linux

IMPORTANT: Before upgrading to OpenText eDirectory 9.3.0, ensure that all the replica or master servers are on 9.2.8 or 9.2.9 version of OpenText eDirectory. To upgrade any of these servers, install the EBA bridge patch bundled with OpenText eDirectory 9.3.0 build. OpenText eDirectory 9.3.0 does not start if schema extension is not applied before applying the bridge patch.

When OpenText eDirectory 9.3.0 is installed on a new server and added into a replica ring where the master server is OpenText eDirectory 9.2.8 or 9.2.9 and EBA is enabled, apply the EBA bridge patch to the OpenText eDirectory 9.2.8 or 9.2.9 servers prior to installing OpenText eDirectory 9.3.0.

- **To install the bridge patch on Linux for a root user, follow the below steps**.

1 Download and untar the OpenText eDirectory build and navigate to the untared location `<Untarred_edirectory_location>/eDirectory/eba_bridge_patch` and execute the following steps. After untaring, locate the `eba_bridge.sch` in this location. Extend the bridge patch schema using below command -

```
ndssch [-h <hostname>[:<port>]][-t <treename>] <admin-FDN> <schemafile>
```

For example:

```
ndssch  -h 10.10.10.10:524 -t tree1 admin.novell eba_bridge.sch
```

2 To apply the bridge patch login as a root user, run the following command-

```
rpm -ivh --force novell-eba-9.2.8.0000.x86_64.rpm
```

3 Restart the `ndsd` services.

- **To install the bridge patch on Linux for a non-root user, follow the below steps**.

1 Download and untar the OpenText eDirectory build and navigate to the untared location `<Untarred_edirectory_location>/eDirectory/eba_bridge_patch` and execute the following steps. After untaring, locate the `eba_bridge.sch` in this location. Extend the bridge patch schema using below command -

```
ndssch [-h <hostname>[:<port>]][-t <treename>] <admin-FDN> <schemafile>
```

For example:

```
ndssch  -h 10.10.10.10:524 -t tree1 admin.novell eba_bridge.sch
```

2 Copy the below library files in `/home/nuser/eDirectory/opt/novell/eDirectory/lib64`., where `/home/nuser` is a user specific path.

```
libebacli.so, libebasrv.so, libebassl_cli.so, libebassl_srv.so and
libgssapi.so
```

3 Restart the `ndsd` services.

NOTE: OpenText eDirectory does not start if schema extension is not applied before applying the bridge patch.

Installing Bridge Patch to Multiple Replica Servers with EBA in Windows

To install the bridge patch on Windows, follow the below steps

1. Unzip the OpenText eDirectory build located at `<extracted_patch>eDirectory\eba_bridge_patch`
2. Update the `upgrade.ni` file present in preUpgrade_930 folder. Mention the tree name, server name and admin credential of the existing eDirectory server in the `upgrade.ni` response file. Below is a sample `upgrade.ni` response file shown with the upgrade configuration.

   ```
   [NWI:NDS]
   mode=configure
   Tree Name=enewtree
   Server Name=enewserver
   Server Container=org
   Admin Context=org
   Admin Login Name=Admin
   Admin Password=env:PASSWORD_VAR
   Require IPV6=NO
   Require EBA=NO
   Require FIPS for TLS=YES
   LDAP TCP Port=389
   LDAP SSL Port=636
   Require TLS=Yes
   Require SS=Yes
   Existing Server=10.10.10.10
   Existing Server Port=524
   [NWI:SNMP]
   Stop service=No
   [NWI:NMAS]
   Methods=CertMutual,Challenge Response,DIGEST-MD5,NDS,Simple
   Password,SAML
   ```

3. Run the below command from powershell.

   ```
   preUpgrade.ps1 -rfile <path_for_Configuration>/upgrade.ni
   ```

4. Stop OpenText eDirectory services before copying these DLLs: `ebacli.dll`, `ebai18n.dll`, `ebaldap.dll`, `ebassl_cli.dll`, `ebasym.dll`, from `<extracted_patch>eDirectory\eba_bridge_patch to C:\NetIQ\eDirectory` (OpenText eDirectory binary location).
5. Start the `ndsd` services.

Installing Bridge Patch to Multiple Replica Servers with EBA in Container

To install the bridge patch on container, follow the below steps.

1. Download and untar the OpenText eDirectory build and navigate to the untarred location `<Untarred_edirectory_location>/eDirectory_930/eba_bridge_patch` and execute the following steps.
2. After untarring,
   - **2a** Copy `eba_brdge.sch` to the container and execute the following command.

     ```
     docker cp eba_brdge.sch <eDirectory Container name>:/config
     ```

   - **2b** Copy `novell-eba-9.2.8.0000.x86_64.rpm` to the container, execute below command.

     ```
     docker cp novell-eba-9.2.8.0000.x86_64.rpm <eDirectory Container name>:/config
     ```

3. Login to the container, navigate to the directory where you copied the schema file in Step 2a on page 26, and run the below command.

   Extending Schema

   ```
   ndssch [-h <hostname>[:<port>]][-t <treename>] <admin-FDN> <schemafile>
   ```

   For example:

   ```
   ndssch  -h 10.10.10.10:524 -t tree1 admin.novell eba_bridge.sch
   ```

4. To install the EBA bridge patch RPM, execute below steps.
   - **4a** Login to the container as a root user.

     ```
     docker exec -it -u root <eDirectory Container name> /bin/bash
     ```

   - **4b** Navigate to the directory where you copied the rpm in Step 2b on age 146, and run the below command.

     ```
     rpm -ivh --force novell-eba-9.2.8.0000.x86_64.rpm
     ```

5. To restart ndsd services, restart OpenText eDirectory container using the below command.

   ```
   docker restart <eDirectory Container name>
   ```

For More Information

For more information on any of the features discussed in this chapter, see the following documentation:

- *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*
- On Linux: check man pages for `nds-install` `ndsconfig`, and `ndscheck`.

  Example: `man nds-install`

3 Installing or Upgrading OpenText eDirectory CE 24.4 (v9.3) on Linux

Use the following information to install or upgrade OpenText eDirectory CE 24.4 (v9.3) on a Linux server:

- "System Requirements" on page 27
- "Prerequisites" on page 29
- "Hardware Requirements" on page 33
- "Forcing the Backlink Process to Run" on page 33
- "Adding the edirAdmin User Group" on page 33
- "Upgrading OpenText eDirectory" on page 34
- "Installing OpenText eDirectory" on page 40

System Requirements

You must install OpenText eDirectory on one of the following 64-bit platforms, at a minimum:

- Memory
  - 2 GB of disk space for the OpenText eDirectory server
  - 2 GB of disk space for every 50,000 users
- Virtualization Systems
  - VMWare ESXi
- One of the following operating systems:

  The following table contains a list of the certified and supported server operating systems that the Identity Vault can run on.

  IMPORTANT: Certified means the Operating System has been fully tested and supported. However, if an Operating System is listed as Supported it means that it has not yet been tested, but it is expected to work.

Certified Server Operating System Version	Supported Operating Systems	Notes
SUSE Linux Enterprise Server 15 SP3, SP4, SP5 and SP6	Supported on later versions of support packs	For the most recent information about the system requirements, see the Release Notes.

Certified Server Operating System Version	Supported Operating Systems	Notes
RHEL8.8 and above	Supported on later versions of support packs	For the most recent information about the system requirements, see the Release Notes. **NOTE:** ◆ You must set SELinux to permissive mode on RHEL 8.8 and above.
RHEL 9.2 and above	Supported on later versions of support packs	For the most recent information about the system requirements, see the Release Notes. **NOTE:** ◆ You must set `SELinux` to permissive mode on RHEL9.2 and above. ◆ If `SELNIUX` is set as `ENFORCING` follow the steps given in "Prerequisites" on page 29.

To determine the version of SUSE Linux you are running, see the `/etc/os-release` file.

Ensure that the latest `glibc` patches (both 32-bit and 64-bit) are applied from Red Hat Errata (http://rhn.redhat.com/errata) on Red Hat systems. The minimum required version of the `glibc` library is version 2.4.

Ensure that an appropriate entry for the Linux machine is available in the `/etc/hosts` file.

NOTE:

- B-tree file system (BTRFS) is not supported with OpenText eDirectory.

Determining the version of OpenText eDirectory

To determine the version of OpenText eDirectory, follow one of the steps mentioned below:

- Run ndsstat.

  The ndsstat utility displays information related to OpenText eDirectory servers, such as the OpenText eDirectory tree name, the fully distinguished server name, and the OpenText eDirectory version. In the following example, OpenText eDirectory 9.3 is the product version (marketing string), and 40201.12 is the binary version (internal build number).

```
osg-dt-srv17:/>ndsstat
Tree Name: EDR93E
Server Name: .CN=Master.O=server.T=EDR93E.
Binary Version: 40301.00
Root Most Entry Depth: 0
Product Version: eDirectory for Linux x86_64 v9.3 [DS]
```

For information on running ndsstat, see "OpenText eDirectory Linux Commands and Usage" in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide, or the ndsstat man page (`ndsstat.1m`).

- Run `ndsd --version`.

  For information on running ndsd, see "OpenText eDirectory Linux Commands and Usage" in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide, or the ndsd man page (`ndsd.1m`).

- Run iMonitor.

  On the Agent Summary page, click Known Servers. Then under Servers Known to Database, click Known Servers. The Agent Revision column displays the internal build number for each server. For example, an Agent Revision number for OpenText™ eDirectory CE 24.4 (v9.3) might be 40301.00.

  For information on running iMonitor, see "Accessing OpenText iMonitor" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

- Run `rpm -qi NDSserv`.

  Entering this command will display similar information to `ndsd --version`.

Prerequisites

IMPORTANT: Check the currently installed OpenText and Third Party applications to determine if those products are supported on OpenText eDirectory 9.3 before upgrading your existing OpenText eDirectory environment. The prerequisites for other OpenText products can be found on the OpenText Documentation site (http://www.netiq.com/documentation/). We also recommend you back up an OpenText eDirectory instance before performing any upgrades on that instance.

- ❒ Ensure that you install the following RPMs based on your operating system:
  - **RHEL 8.x and RHEL9:** `dnf-utils` and `createrepo`

    Execute the following steps to install RHEL 8.x RPM

    1. Install the `yum-utils` package

       yum install `createrepo yum-utils`
    2. Install the following libraries required for OpenText eDirectory.

       yum install libgcc*.i686 libncurses*
    3. Navigate to the `SELinux` configuration file located in `/etc/selinux/config` location and set `SELinux` to permissive mode as follows:

       `SELINUX=permissive`
  - **RHEL Minimal Versions:** `ncurses, libxcrypt-compat, iproute, initscripts, procps,` and `net-tools`.
  - **SLES:** Zypper
- ❒ If upgrading the RHEL OS fails when OpenText eDirectory is installed follow the below steps as a work around.
  - Use the below command to remove the package before the RHEL OS upgrade.

    `rpm -e --nodeps novell-NLDAPsdk`
  - Upgraded the RHEL OS from 8.x to 9.x and later.

- After the OS upgrade, install the package using the below command.

  `rpm -ivh ./novell-NLDAPsdk*`

- Restart the OpenText eDirectory service.

❒ Execute the following steps to install OpenText eDirectory, if SELINUX is set to Enforcing:

1. Download and untar the OpenText eDirectory build.
2. Install the OpenText eDirectory packages, run `./nds-install` located at `untared_location/eDirectory/setup`.
3. Before configuring the tree, create data directory at default OpenText eDirectory dib location `/var/opt/novell/eDirectory`.

   Now, dib location will be `/var/opt/novell/eDirectory/data`

   (For custom location provide custom path `ex/home/edirectory/data`)
4. Run following commands,
   - `touch /var/opt/novell/eDirectory/data/ndsd.pid`
   - `semanage fcontext -a -t var_run_t '/var/opt/novell/eDirectory/data/ndsd.pid'`
   - `restorecon -v '/var/opt/novell/eDirectory/data/ndsd.pid'`
5. Now configure the tree using `ndsconfig` new command.

❒ (Conditional) Novell International Cryptographic Infrastructure (NICI) 3.3 and OpenText eDirectory 9.3 support key sizes up to 8192 bits. If you want to use a 8 KB key size, every server must be upgraded to OpenText eDirectory 9.3. In addition, every workstation using the management utilities, for example, Identity Console must have NICI 3.3 installed on it.

When you upgrade your Certificate Authority (CA) server to OpenText eDirectory 9.3, the key size will not change but will still be 2 K. The only way to create a 8 K key size is recreate the CA on an OpenText eDirectory 9.3 server. In addition, you would have to change the default from 2 K to 8 K for the key size during the CA creation.

When you install OpenText eDirectory, the nds-install utility automatically installs NICI. For more information about installing OpenText eDirectory, see “Using the nds-install Utility to Install OpenText eDirectory Components” on page 41. However, if you need to install only NICI, and not OpenText eDirectory itself, on a workstation that has the management utilities installed, you must install NICI manually. For more information about manually installing NICI, see “Installing NICI” on page 45.

❒ Ensure to obtain 168-bit 3DES tree key for your OpenText eDirectory servers.

❒ (Conditional) Service Location Protocol (SLP) should be installed and configured only if you plan to use SLP to resolve tree names when DNS is not available.

With OpenText eDirectory 9.3, SLP does not get installed as part of the OpenText eDirectory installation.

Only a root user can install SLP.

For more information on installing SLP, refer to “Using SLP with OpenText eDirectory” on page 40.

❒ The Linux host enabled for multicast routing

To check if the host is enabled for multicast routing, enter the following command:

`/bin/netstat -nr`

The following entry should be present in the routing table:

```
224.0.0.0 0.0.0.0
```

If the entry is not present, log in as root and enter the following command to enable multi-cast routing:

```
route add -net 224.0.0.0 netmask 240.0.0.0 dev interface
```

The *interface* could be a value such as eth0, hme0, hme1, or hme2, depending on the NIC that is installed and used.

For more information on multi-cast and broadcast routes, refer to the OpenSLP Web site (http://www.openslp.org/doc/html/UsersGuide/Installation.html).

- ❒ Network server time synchronized

  Use Network Time Protocol's (NTP) ntp to synchronize time across all network servers.

- ❒ (Conditional) If you are installing a secondary server, all the replicas in the partition that you install the product on should be in the On state.

- ❒ (Conditional) If you are installing a secondary server into an existing tree as a non-administrator user, create a container and then partition it. Ensure that you have the following rights:

  - Supervisor rights to this partition.
  - All Attributes rights: read, compare, and write rights over the W1.KAP.Security object.
  - Entry rights: browse rights over Security container object.
  - All Attributes rights: read and compare rights over Security container object.
  - (Conditional) If the W1.KAP.Security object exists, all attributes rights: read, compare, and write rights over this object. For more information about the W1.KAP.Security object, see Creating an AES 256-Bit Tree Key in the *NICI Administration Guide*.

- ❒ (Conditional) If you are installing a secondary server into an existing tree as a non-administrator user, ensure that at least one of the servers in the tree has the same or higher OpenText eDirectory version as that of the secondary being added as container admin. In case the secondary being added is of later version, then the schema needs to be extended by the administrator of the tree before adding the secondary using container admin.

- ❒ While configuring OpenText eDirectory, you must enable SLP services and a NetWare Core Protocol (NCP) port (the default is 524) in the firewall to allow the secondary server addition. Additionally, you can enable the following service ports based on your requirements:

  - LDAP clear text - 389
  - LDAP secured - 636
  - HTTP clear text - 8028
  - HTTP secured - 8030

  In case, if you have enabled user-defined ports, you must mention these ports while configuring OpenText eDirectory.

  NOTE: This step is required only if you have SLP configured in your system.

- ❒ Do not set the user-defined ports to 8008 and 8010 while upgrading OpenText eDirectory 8.8 SP8 or later versions to 9.2. If the ports are set to 8008 or 8010, `ndsconfig` assumes that the server is a pre-eDirectory 8.8.*x* server and automatically resets them to 8028 and 8030 respectively.

- During OpenText eDirectory upgrade, if SecretStore has not already been configured with the previous versions, or you do not want to configure SecretStore, use the `-m no_ss` option with the `nds-install` utility.
- The OpenText eDirectory Management Toolbox (eMBox) lets you access all of the OpenText eDirectory back-end utilities remotely, as well as on the server. The command line client is a Java application. To run it, you must install the latest version of Oracle Java (1.8 or above). You must also ensure to upgrade any older version of Java by installing the patch upgrades available. Once you have the latest version of Java installed, export any of the following environment variables:
  - `EDIR_JAVA_HOME`
  - `JAVA_HOME`
  - `JRE_HOME`

  NOTE:
  - If none of the above mentioned environment variables are found, command line client searches for the Java binary in the default PATH environment variable.
  - If you are using any prior version of OpenText eDirectory 9.0 SP4, To run the command line client, you must have access to the Java Runtime Environment, Oracle Java 1.8, which is installed with OpenText eDirectory.

RPM Signing: Public Key to validate the signature

Use the following steps to perform the RPM Signature Verification, before installing the OpenText eDirectory components on Linux systems from OpenText eDirectory 9.2.7 and above onwards:

1. Navigate to the following location for Public Key:

   ```
   <untarred location of eDirectory>/eDirectory/license/
   MicroFocusGPGPackageSign.pub
   ```

2. Run the following command to import the Public Key:

   ```
   rpm --import MicroFocusGPGPackageSign.pub
   ```

3. (Optional) Run the following command to verify the RPM signature:

   ```
   rpm --checksig -v <RPM Name>
   ```

   For Example:

   ```
   rpm --checksig -v novell-NDSbase-9.2.x.0000.x86_64.rpm
   ```

NOTE: OpenText eDirectory 9.3.0 be installed on FIPS enabled OS with the supported versions of RHEL8 or later and RHEL 9 or later, SLES 12 SP5 and SLES 15 SP3, SP4, SP5 and SP6.

Configuring Static IP Address

Static IP address must be configured on the server for the OpenText eDirectory to perform efficiently. Configuring OpenText eDirectory on the servers with DHCP address can lead to unpredictable results.

Hardware Requirements

Hardware requirements depend on the specific implementation of OpenText eDirectory. Two factors increase performance: more cache memory and faster processors. For best results, cache as much of the Directory Information Base (DIB) Set as the hardware allows.

OpenText eDirectory scales well on a single processor. However, OpenText eDirectory 9.3 takes advantage of multiple processors. Adding processors improves performance in some areas — for example, logins — and having multiple threads active on multiple processors also improves performance. OpenText eDirectory itself is not processor intensive, but it is I/O intensive.

The following table illustrates typical system requirements for OpenText eDirectory for Linux:

Objects	Memory	Hard Disk
100,000	2+ GB	300 MB
1 million	4 GB	1.5 GB
10 million	4+ GB	15 GB

Forcing the Backlink Process to Run

Because the internal OpenText eDirectory identifiers change when upgrading to OpenText eDirectory, the backlink process must update backlinked objects for them to be consistent.

Backlinks keep track of external references to objects on other servers. For each external reference on a server, the backlink process ensures that the real object exists in the correct location and verifies all backlink attributes on the master of the replica. The backlink process occurs two hours after the database is open, and then every 780 minutes (13 hours). The interval is configurable from 2 minutes to 10,080 minutes (7 days).

After migrating to OpenText eDirectory, start the DSTrace process by issuing the `ndstrace -l>log&` command, which runs the process at the background. This allows you to properly analyze the results of the backlinker process, which takes 4 to 10 minutes. Then force the backlink process to run by issuing the `ndstrace -c 'set ndstrace=*B'` command from the DSTrace OS command prompt. Review the results of the log file created in the first step. Then you can unload the DSTrace process by issuing the `ndstrace -u` command. Running the backlink process is especially important on servers that do not contain a replica.

Adding the edirAdmin User Group

In OpenText eDirectory 9.3 release, eDirAdmin Fixed Group name has been removed. It has been modified to incorporate the current behavior. It's flexible, configurable, modifiable and is optional (not mandatory). New installation will not create edirAdmin group. It will give an option to enter a group name, if given it will be used in place of edirAdmin. Group name can be set using -g option in `ndsconfig new/add/upgrade` option. User has to enter a valid and existing group name in `-g` option. There should not be a gap between -g and groupname (-g <groupname>). The group name given will be stored in configuration file as "`n4u.server.osgroup-name`". If empty, it will not be used, otherwise used to set restrictions on selected directories. The user which creates the instance,

should be part of the group to set the restrictions. It is optional and if not given, it is assumed that no group setting has to be made. For upgrade/migration from 9.2.5/9.2.6, edirAdmin is already the owner, the configuring user has to be part of that group as well.

NOTE: The following edirAdmin User Group settings are applicable for 9.2.5 and 9.2.6 versions of OpenText eDirectory.

In OpenText eDirectory 9.2.5 release, a new edirAdmin user group is added to OpenText eDirectory during installation. This user group has access permission to the directories where the data, log, and configuration files are installed. Any user added to this group will have the group permissions assigned to these directories.

During root installation and configuration, the installer creates the edirAdmin user group by default. However, for non-root installation and configuration, this group must be manually created before upgrading or installing the latest version of the OpenText eDirectory. After creating the group, you must ensure that user is added to this group to be able to able to configure OpenText eDirectory server instances using his or her individual tarball installation, or by using a binary installation. On the other hand, when a non-root user performs configuration for root installed OpenText eDirectory, the group is added as part of installation. You must ensure that the user is added to this group, else they might encounter errors while configuring OpenText eDirectory using the `ndsconfig` utility.

On the server where OpenText eDirectory is installed, open a terminal and run the following commands:

- To add the edirAdmin group, run `groupadd edirAdmin`
- To add a user to the edirAdmin group, run `usermod -a -G edirAdmin <username>`

  Where `<username>` is the name of the user. For example, to add the user John to the edirAdmin group, you must execute the command:

  ```
  $ usermod -a -G edirAdmin john
  ```

NOTE: Before adding the user to the OpenText edirAdmin user group, the administrator must ensure that the user is not logged in to the server through any terminal session. After adding, the user can login and configure OpenText eDirectory using the ndsconfig utility.

Upgrading OpenText eDirectory

It is recommended not to directly upgrade OpenText eDirectory from version 8.8SP1 to 9.3.You must upgrade to 9.2 and then to 9.3. Upgrading OpenText eDirectory from any previous version of 8.8 SP1 to 9.2 impacts the performance of your OpenText eDirectory server.

NOTE: To upgrade from a 32-bit version of OpenText eDirectory to a 64-bit version of eDirectory, first upgrade 32-bit version to eDirectory 8.8.x 64-bit version and then upgrade it to OpenText eDirectory 9.2. You can follow the same procedure for upgrading a 64-bit OpenText eDirectory to OpenText eDirectory 9.3.

The following sections provide information to help you upgrade your existing OpenText eDirectory installation to the current version.

- "Server Health Checks" on page 35
- "Upgrading on Linux Servers Other Than OES" on page 35
- "Unattended Upgrade of OpenText eDirectory on Linux" on page 36
- "Upgrading the Tarball Deployment of eDirectory 9.3" on page 37
- "Non-Root Users Upgrading OpenText eDirectory 9.3" on page 38
- "Upgrading Multiple Instances" on page 39

NOTE: The `ndsconfig upgrade` command is used to upgrade the necessary configuration of the individual components such as HTTP, LDAP, SNMP, SAS, and NetIQ Modular Authentication Service (NMAS).

Server Health Checks

With OpenText eDirectory 9.3, when you upgrade OpenText eDirectory, a server health check is conducted by default to ensure that the server is safe for the upgrade:

- "Partitions and Replica Health" on page 189

Based on the results obtained from the health checks, the upgrade will either continue or exit as follows:

- If all the health checks are successful, the upgrade will continue.
- If there are minor errors, the upgrade will prompt you to continue or exit.
- If there are critical errors, the upgrade will exit.

See Appendix B, "OpenText eDirectory Health Checks," on page 187 for a list of minor and critical error conditions.

Skipping Server Health Checks

To skip server health checks, run `nds-install -j` or `ndsconfig upgrade -j` from the installation folder.

For more information, see Appendix B, "OpenText eDirectory Health Checks," on page 187.

Upgrading on Linux Servers Other Than OES

OpenText eDirectory upgrade is supported from OpenText eDirectory 9.20 onwards.

To upgrade, use the nds-install utility. This utility is located in the `Setup` directory of the downloaded file for Linux platform. Enter the following command from the `Setup` directory:

```
./nds-install
```

After the upgrade to OpenText eDirectory 9.2, the default location of the configuration files, data files, and log files are changed to `/etc/opt/novell/eDirectory/conf`, `/var/opt/novell/eDirectory/data`, and `/var/opt/novell/eDirectory/log`, respectively.

The new directory `/var/opt/novell/eDirectory/data` uses a symbolic link to the `/var/nds` directory.

The old configuration file `/etc/nds.conf` is migrated to `/etc/opt/novell/eDirectory/conf` directory. The old configuration file `/etc/nds.conf` and the old log files under `/var/nds` are retained for reference.

NOTE: Run `ndsconfig upgrade` after `nds-install`, if the upgrade of the DIB fails and `nds-install` asks to do so. If OpenText eDirectory services are not starting after upgrading the OS from RHEL 6.8 to 7.1, run the `ndsconfig upgrade` command.

NOTE: Health check fails due to time sync. To resolve this issue, perform a time sync between the instances. You can ignore this warning message during upgrade.

Unattended Upgrade of OpenText eDirectory on Linux

On Linux, OpenText eDirectory provides switches and options along with the install script and configuration utility that facilitates the unattended upgrade. The following sections discuss various steps for unattended OpenText eDirectory upgrade on Linux:

1. Perform the health check of OpenText eDirectory:

   Health check of all the root instances planned for upgrade is manually done by using `ndscheck` utility.

   1a `export LD_LIBRARY_PATH to the <untarred location of eDirectory>/eDirectory/setup/utils`

   1b Run ndscheck using one of the below commands:

   ```
   <untarred location of eDirectory>/eDirectory/setup/utils/ndscheck -a <user name> -w passwd --config-file <nds.conf with absolute path>
   ```

   Passing the password through environment variable: `<untarred location of 88SP8>/eDirectory/setup/utils/ndscheck -a <user name> -w env:<environment variable> --config-file <nds.conf with absolute path>`

   Passing the password through file: `<untarred location of 88SP8>/eDirectory/setup/utils/ndscheck -a <user name> -w file:<filename> --config-file <nds.conf with absolute path>`

   Any one of the above can be used in the automated script for the health check. For example:

   ```
   /Builds/eDirectory/utils/ndscheck -a admin.novell -w n
   /Builds/eDirectory/utils/ndscheck -a admin.novell -w env:ADM_PASWD
   /Builds/eDirectory/utils/ndscheck -a admin.novell -w file:adm_paswd
   ```

2. Upgrade the OpenText eDirectory 9.3 packages:

   2a Run the nds-install script to upgrade the packages as below:

   ```
   nds-install -u -i -j
   ```

3. Update the following environment variables:

```
PATH=/opt/novell/eDirectory/bin:/opt/novell/eDirectory/sbin:$PATH
LD_LIBRARY_PATH=/opt/novell/eDirectory/lib:/opt/novell/eDirectory/lib/
nds-modules:/opt/novell/lib:$LD_LIBRARY_PATH
MANPATH=/opt/novell/man:/opt/novell/eDirectory/man:$MANPATH
TEXTDOMAINDIR=/opt/novell/eDirectory/share/locale
```

4 Upgrade OpenText eDirectory using the ndsconfig utility for all the root instances by executing the following commands:

```
ndsconfig upgrade -a <user name> -w passwd -c --config-file <nds.conf with
absolute path> --configure-eba-now <yes/no>
```

NOTE: To enable Enhanced Background Authentication, specify *yes* to the `--configure-eba-now` switch in the ndsconfig upgrade command. Otherwise, specify *no* to configure it later.

Passing the password through environment variable: `ndsconfig upgrade -a <user name> -w env:<environment variable> -c --config-file <nds.conf with absolute path> --configure-eba-now <yes/no>`

Passing the password through file: `ndsconfig upgrade -a <user name> -w file:<filename with absolute/relative path> -c --config-file <nds.conf with absolute path> --configure-eba-now <yes/no>`

Any of the above can be used in the automated script for the eDirectory upgrade. For example:

```
ndsconfig upgrade -a admin.novell -w n -c -–config-file /etc/opt/
novell/eDirectory/conf/nds.conf --configure-eba-now <yes/no>
```

```
ndsconfig upgrade -a admin.novell -w env:ADM_PASWD -c --config-file /
etc/opt/novell/eDirectory/conf/nds.conf --configure-eba-now <yes/no>
```

```
ndsconfig upgrade -a admin.novell -w <password file path>/adm_paswd -c
--config-file /etc/opt/novell/eDirectory/conf/nds.conf --configure-eba-
now <yes/no>
```

Upgrading the Tarball Deployment of eDirectory 9.3

If you want to upgrade the tarball deployment from OpenText eDirectory 9.20 to OpenText eDirectory 9.3, perform the following steps:

1 Download the tarball build.

2 Take backup of the following configuration files:

- `$NDSHOME/eDirectory/etc/opt/novell/eDirectory/conf/ndsimon.conf`
- `$NDSHOME/eDirectory/etc/opt/novell/eDirectory/conf/ice.conf`
- `$NDSHOME/eDirectory/etc/opt/novell/eDirectory/conf/ndsimonhealth.conf`
- `$NDSHOME/eDirectory/etc/opt/novell/eDirectory/conf/ndssnmp/ndssnmp.cfg`
- $NDSHOME is the location where OpenText eDirectory is installed.

3 For upgrade of OpenText eDirectory versions lower than 8.8 SP1, do the following:

- Perform disk space check using `ndscheck -D --config-file conf_file_path`

- Create an empty file `upgradeDIB` under the DIB location of each server instance.

  The list of instances can be obtained using the `ndsmanage` utility.

4. Run pre upgrade health check for the all instances using `ndscheck` and check the `ndscheck.log` file for any errors before proceeding with the upgrade.
5. Stop all instances using `ndsmanage`.
6. Untar the tarball in the same location (`$NDSHOME`) where OpenText eDirectory is installed. By untarring the tarball in the same location, you are overwriting the binaries and libraries.
7. Upgrade the following package if necessary.

Platform	Command	Packages
Linux		◆ `novell-NOVLsubag-9.2.0-0.x86_64.rpm` ◆ `netiq-openssl-3.0.15.x86_64.rpm` **NOTE:** For more information on installing 64-bit NICI, refer to the "Installing NICI" on page 45.

8. Restore the configuration files.
9. Run the `$NDSHOME/eDirectory/opt/novell/eDirectory/bin/ndspath` for setting all environment variables.
10. Run `ndsconfig upgrade -j` for all instances. While running `ndsconfig upgrade` follow the order in which the master replica is the first and followed by Read/Write and others.

Non-Root Users Upgrading OpenText eDirectory 9.3

A non-root user can upgrade OpenText eDirectory using the new version of the tarball. Perform the following steps to upgrade OpenText eDirectory as a non-root user:

1. Login to the machine where OpenText eDirectory is installed as non-root user using ssh.
2. Go to the directory where you want to install OpenText eDirectory.
3. Untar the new version of the tar file as follows:

   ```
   tar xvf /tar_file_name
   ```

   The `etc`, `opt`, and `var` directories are created.
4. Stop all eDirectory servers as follows:

   ```
   ndsmanage stopall
   ```

5. Upgrade OpenSSL and NICI as a root user using the following commands:
   - OpenSSL should be upgraded before you upgrade the NICI. To install OpenSSL, enter the following command::

     ```
     rpm -Uvh openssl_rpm_absolute_path/netiq-openssl-3.0.15.x86_64.rpm
     ```

   - To upgrade 64-bit NICI, enter the following command

```
rpm -Uvh NICI_rpm_absolute_path/nici64-3.3.0-0.00.x86_64.rpm
```

- To ensure that NICI is set to server mode, enter the following as root user:

  ```
  /var/opt/novell/nici/set_server_mode64
  ```

6 Run the following command to upgrade the OpenText eDirectory servers:

```
ndsconfig upgrade
```

Upgrading Multiple Instances

This section contains the following information:

- "Root User has Multiple Instances" on page 39
- "Non Root User's Instances" on page 39
- "Order of Upgrade" on page 39

Root User has Multiple Instances

If you run nds-install after upgrading the package, it prompts you to upgrade the DIB files of all the OpenText eDirectory server instances, which might take a long time to complete. If you wish to perform the DIB upgrade in parallel, you can do it manually. For information about manually upgrading the DIB, refer to the OpenText eDirectory Release Notes. If you upgrade the DIB for all the active instances one by one, it runs the `ndsconfig upgrade` command separately for each instance. If you have a larger DIB, you can select **No** and run the `ndsconfig upgrade` in parallel in separate shells, which can reduce the upgrade time of each instance.

Non Root User's Instances

If you have non root users' instances which are using root users' binaries, before doing the package upgrade you need to run ndscheck for such instances and make sure that their health is proper by referring the `ndscheck.log` file. If you run nds-install, it stops all the instances, including the non root user's instances. After doing the package upgrade, the `nds-install` command does not call `ndsconfig upgrade` for non-root user's instances. You need to manually run `ndsconfig upgrade` for all non-root user's instances to start these instances.

Order of Upgrade

While running `ndsconfig upgrade`, it is recommended to follow the order in which master replica comes first and then Read/Write or other replicas.

NOTE: To upgrade the Replica Server with EBA in Linux for eDirectory 9.2.8 and 9.2.9, see "Installing Bridge Patch to Multiple Replica Servers with EBA in Linux" on page 24.

Installing OpenText eDirectory

The following sections provide information about installing OpenText eDirectory on Linux:

- "Using SLP with OpenText eDirectory" on page 40
- "Using the nds-install Utility to Install OpenText eDirectory Components" on page 41
- "Non-root User Installing OpenText eDirectory 9.3" on page 44
- "Non-Root User Configuring OpenText eDirectory with Root Install" on page 48
- "Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server" on page 49
- "Using ndsconfig to Configure Multiple Instances of OpenText eDirectory 9.3" on page 56
- "Using ndsconfig to Install a Linux Server into a Tree with Dotted Name Containers" on page 63
- "Using the nmasinst Utility to Configure NMAS" on page 63
- "Non-root User SNMP Configuration" on page 64
- "Locating Log Files" on page 65

Using SLP with OpenText eDirectory

In earlier releases of OpenText eDirectory, SLP was installed during the OpenText eDirectory install. But with eDirectory 9.1 and above, you need to separately install SLP before proceeding with the OpenText eDirectory install.

If you plan to use SLP to resolve tree names, you should install and configure the protocol, and the SLP directory agents (DAs) should be stable.

1. Install OpenSLP, if it is not already installed.
2. Follow the on-screen instructions to complete the SLP installation.
3. Start SLP manually as follows:

   ```
   /etc/init.d/slpd start
   ```

For more information, refer to Appendix C, "Configuring OpenSLP for OpenText eDirectory," on page 193.

Similarly, when you uninstall the SLP package, you need to stop SLP manually, as follows:

```
/etc/init.d/slpd stop
```

If you don't want to (or cannot) use SLP, you can use the flat file `hosts.nds` to resolve tree names to server referrals. The `hosts.nds` file can be used to avoid SLP multicast delays when SLP DA is not present in the network.

`hosts.nds` is a static lookup table used by OpenText eDirectory applications to search OpenText eDirectory partition and servers. In the `hosts.nds` file, for each tree or server, a single line contains the following information:

- Tree/Server Name: Tree names end with a trailing dot (.).
- Internet Address: This can be a DNS name or IP address.
- Server Port: Optional, appended with a colon (:) to the Internet address.

Local server need not have an entry in this file unless it is listening on non-default NCP port.

The syntax followed in the `hosts.nds` file is as follows:

```
<[partition name.]tree name>.  <host-name/ip-addr>[:<port>]
<server name>  <dns-addr/ip-addr>[:<port>]
```

For example:

```
# This is an example of a hosts.nds file:
# Tree name             Internet address/DNS Resolvable Name
  CORPORATE.            myserver.mycompany.com
  novell.CORPORATE.     1.2.3.4:524

# Server name           Internet address
  CORPSERVER            myserver.mycompany.com
```

See the `hosts.nds` man page for more details.

If you decide to use SLP to resolve the tree name to determine if the eDirectory tree is advertised, after eDirectory and SLP are installed, enter the following:

```
/usr/bin/slptool findattrs services:ndap.novell///(svcname-ws==[treename
or *])"
```

For example, to search for the services whose `svcname-ws` attribute match with the value SAMPLE_TREE, enter the following command:

```
/usr/bin/slptool findattrs services:ndap.novell///(svcname-
ws==SAMPLE_TREE)/"
```

If you have a service registered with its `svcname-ws` attribute as SAMPLE_TREE, then the output will be similar to the following:

```
service:ndap.novell:///SAMPLE_TREE
```

If you do not have a service registered with its `svcname-ws` attribute as SAMPLE_TREE, there will be no output.

For more information, see Appendix C, “Configuring OpenSLP for OpenText eDirectory,” on page 193.

Using the nds-install Utility to Install OpenText eDirectory Components

Use the nds-install utility to install OpenText eDirectory components on Linux systems. This utility is located in the `Setup` directory of the downloaded file for the Linux platform. The utility adds the required packages based on what components you choose to install.

1. Enter the following command at the setup directory:

   ```
   ./nds-install
   ```

   If you do not provide the required parameters in the command line, the nds-install utility will prompt you for the parameters.

   The following table provides a description of the nds-install utility parameters:

nds-install Parameter	Description
`-h` or `--help`	Displays help for nds-install.
`-i`	Prevents the nds-install script from invoking the `ndsconfig upgrade` command if a DIB is detected at the time of the upgrade.
`-j`	Jumps or overrides the health check option before installing OpenText eDirectory. For more information about health checks, refer to Appendix B, “OpenText eDirectory Health Checks,” on page 187.
`-m`	Specifies the module name to configure. While configuring a new tree, you can configure only the ds module. After configuring the ds module, you can add the NMAS, LDAP, SAS, SNMP, HTTP services, and OpenText SecretStore (ss) using the `add` command. If the module name is not specified, all the modules are installed.
`-u`	Specifies the option to use in an unattended install mode.
`-f`	This option is used to force upgrade/downgrade to any version of OpenText eDirectory.

The installation program installs the following RPMs:

eDirectory Component	Packages Installed	Description
eDirectory Server	• novell-NDSbase • novell-NDScommon • novell-NDSmasv • novell-NDSserv • novell-NDSimon • novell-NDSrepair • novell-NDSdexvnt • novell-NOVLsubag • novell-NOVLsnmp • novell-NOVLpkit • novell-NOVLpkis • novell-NOVLpkia • novell-NOVLembox • novell-NOVLlmgnt • novell-NOVLxis • novell-NLDAPsdk • novell-NLDAPbase • novell-NOVLsas • novell-NOVLntls • novell-NOVLnmas • novell-NOVLldif2dib • novell-NOVLncp • novell-eba	The OpenText eDirectory replica server is installed on the specified server.
Administration Utilities	• novell-NOVLice • novell-NDSbase • novell-NLDAPbase • novell-NLDAPsdk • novell-NOVLpkia • novell-NOVLxis • novell-NOVLlmgnt	The OpenText Import Conversion Export and LDAP Tools administration utilities are installed on the specified workstation.

2 If you are prompted, enter the complete path to the license file.

You will be prompted to enter the complete path to the license file only if the installation program cannot locate the file in the default location. The default location is the `/var`, the mounted license diskette, or the current directory.

If the path you entered is not valid, you will be prompted to enter the correct path.

3 After the installation is complete, update and export the following environment variables to use OpenText eDirectory utilities in the current session:

```
export PATH=$PATH:/opt/novell/eDirectory/bin:/opt/novell/eDirectory/sbin

export MANPATH=$MANPATH:/opt/novell/man:/opt/novell/eDirectory/man

export TEXTDOMAINDIR=/opt/novell/eDirectory/share/locale
```

You can use the ndsconfig utility to configure OpenText eDirectory server after installation.

NetIQ Modular Authentication Service (NMAS) is installed as part of the server component. By default, ndsconfig configures NMAS. You can also use the nmasinst utility to configure NMAS server after installation. This must be done after configuring OpenText eDirectory with ndsconfig.

By default, OpenText eDirectory server runs in FIPS mode. To disable the FIPS mode, pass `n4u.server.fips_tls=0` with `ndsconfig set` command and restart the server. For example, `ndsconfig set n4u.server.fips_tls=0`.

When FIPS mode is enabled in your eDirectory environment, all eDirectory applications/modules using OpenSSL will always use OpenSSL in FIPS mode. Operating eDirectory in FIPS mode does not allow communication over SSLv3 and restricts the cipher usage to high strength ciphers. For more information, see Configuring LDAP Objects and Configuring HTTP Server Object in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

For more information on the ndsconfig utility, see "The ndsconfig Utility" on page 157.

For more information on the nmasinst utility, see "Using the nmasinst Utility to Configure NMAS" on page 63.

NOTE: After you install eDirectory, we recommend that you exclude the DIB directory on your OpenText eDirectory server from any antivirus or backup software processes. Use the OpenText eDirectory Backup Tool to back up your DIB directory.

For more information about backing up OpenText eDirectory, see "Backing Up and Restoring OpenText eDirectory," in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Non-root User Installing OpenText eDirectory 9.3

A non-root user can install eDirectory 9.3 using the tarball.

Prerequisites

- ❒ If you want to install OpenText eDirectory using the tarball and not the nds-install utility, make sure that OpenSSL is installed before installing the NICI. For information on installing OpenSSL, refer to "Installing OpenSSL" on page 45.
- ❒ If you want to install OpenText eDirectory using the tarball and not the nds-install utility, ensure that NICI is installed. For information on installing NICI, refer to "Installing NICI" on page 45.
- ❒ Ensure that SNMP subagent is installed using the command `rpm --nodeps <path of snmp subagent rpm>`.
- ❒ If you want to use SLP and SNMP, ensure that they are installed by the root user.

❒ Write rights to the directory where you want to install OpenText eDirectory.

If you are a non-administrator user, ensure that you have the appropriate rights as mentioned in the “Prerequisites” on page 29 section.

Installing OpenSSL

OpenSSL should be installed followed by NICI installation before you proceed with the OpenText eDirectory installation.

Root User Installing OpenSSL

To install OpenSSL, enter the following command:

```
rpm -ivh openssl_rpm_absolute_path/netiq-openssl-3.0.15.x86_64.rpm
```

Installing NICI

NICI should be installed before you proceed with the OpenText eDirectory installation. Because the required NICI packages are used system-wide, we recommend you use the root user to install the necessary packages.

With OpenText eDirectory 9.3, 32 and 64-bit applications can coexist in a single system. Refer to the following sections for installing NICI.

Root User Installing NICI

To install 64-bit NICI, enter the following command:

```
rpm -ivh NICI_rpm_absolute_path/nici64-3.3.0.0000.x86_64.rpm
```

To ensure that NICI is set to server mode, enter the following as root user:

```
/var/opt/novell/nici/set_server_mode64
```

Configuring User Service on SLES 15 and Above

To support services for non-root users on these platforms, start `systemd` specific to the user as a one-time activity.

The following are the advantages of starting services as a non-root user:

- A system administrator can monitor a service.
- The computer starts the service on reboot.

To start `systemd` specific to a user, login to the machine where OpenText eDirectory is installed as root user and run the following command:

```
systemctl start user@<uid>.service
```

where `uid` is the User ID of the user.

For example, `systemctl start user@1001.service`

To enable persistent `systemd` user instance, run the following command:

```
loginctl enable-linger user
```

NOTE: In case you move the `datadir` to a new location after configuring OpenText eDirectory on SLES 15 and above, ensure to perform the following steps:

- Update the new location of the `ndsd.pid` file in the service file found in the `/usr/lib/systemd/system/` location.

  For example, when the `nds.conf` file is originally located at the `/etc/opt/novell/eDirectory`, a sample service file will be created as shown below:

  `/usr/lib/systemd/system/ndsdtmpl-etc-opt-novell-eDirectory-conf-ds.conf@.service`.
- Re-load the daemon by using `systemctl daemon-reload` command.
- Restart the OpenText eDirectory server.

Installing OpenText eDirectory

1. Login to the machine where OpenText eDirectory is installed as non-root user using ssh.
2. Go to the directory where you want to install OpenText eDirectory.
3. Untar the tar file as follows:

   ```
   tar xvf /tar_file_name
   ```

   The `etc` and `opt` directories are created.
4. Export the paths as follows:
   - **Manually export the environment variables by entering the following commands:**

     ```
     export LD_LIBRARY_PATH=custom_location/eDirectory/opt/novell/eDirectory/lib64:custom_location/eDirectory/opt/novell/eDirectory/lib64/nds-modules:custom_location/eDirectory/opt/novell/lib64:$LD_LIBRARY_PATH
     ```

     ```
     export PATH=custom_location/eDirectory/opt/novell/eDirectory/bin:custom_location/eDirectory/opt/novell/eDirectory/sbin:/opt/novell/eDirectory/bin:$PATH
     ```

     ```
     export MANPATH=custom_location/eDirectory/opt/novell/man:custom_location/eDirectory/opt/novell/eDirectory/man:$MANPATH
     ```

     ```
     export TEXTDOMAINDIR=custom_location/eDirectory/opt/novell/eDirectory/share/locale:$TEXTDOMAINDIR
     ```

     Use the ndspath script to export the environment variables by performing the following steps:

     If you do not want to export the paths manually, prefix the ndspath script to the utility.

     - Run the utility you want as follows:

       ```
       custom_location/eDirectory/opt/novell/eDirectory/bin/ndspath utility_name_with_parameters
       ```

- Alternatively, you can export the paths in the current shell by explicitly providing the location of the OpenText eDirectory tarball to ndspath utility:

  ```
  . custom_location/eDirectory/opt/novell/eDirectory/bin/ndspath
  custom_location/eDirectory
  ```

  NOTE: Ensure that you enter the above commands from the *custom_location*/eDirectory/opt directory.

  After entering the above commands, run the utilities as you would normally do.

- Call the script in your profile, bashrc, or similar scripts. Therefore, whenever you log in or open a new shell, you can start using the utilities directly.

5 Configure OpenText eDirectory in the usual manner.

You can configure OpenText eDirectory in any of the following ways:

- Use the ndsconfig utility as follows:

  ```
  ndsconfig new [-t <treename>] [-n <server_context>] [-a <admin_FDN>]
  [-w <admin password>] [-i] [-S <server_name>] [-d <path_for_dib>] [-
  m <module>] [e] [-L <ldap_port>] [-l <SSL_port>] [-o <http_port>] -
  O <https_port>] [-p <IP address:[port]>] [-c] [-b <port_to_bind>] [-
  B <interface1@port1>, <interface2@port2>,..] [-D <custom_location>]
  [--config-file <configuration_file>] [--configure-eba-now <yes/no>]
  ```

  For example:

  ```
  ndsconfig new -t mary-tree -n novell -a admin.novell -S linux1 -d /
  home/mary/inst1/data -b 1025 -L 1026 -l 1027 -o 1028 -O 1029 -D /
  home/mary/inst1/var --config-file /home/mary/inst1/nds.conf --
  configure-eba-now yes
  ```

  The port numbers you enter need to be in the range 1024 to 65535. Port numbers lesser than 1024 are normally reserved for the super-user and standard applications. Therefore, you cannot assume the default port 524 for any eDirectory applications.

  This might cause the following applications to break:

  - The applications that don't have an option to specify the target server port.
  - The older applications that use NCP, and run as root for 524.

- Use the ndsmanage utility to configure a new instance. For more information, refer to the "Creating an Instance through ndsmanage" on page 59.

To enable Enhanced Background Authentication, specify *yes* to the --configure-eba-now switch in the ndsconfig upgrade command. Otherwise, specify *no* to configure it later. If you do not pass the --configure-eba-now switch to the ndsconfig command, OpenText eDirectory prompts you to specify your choice. By default, the configuration is set to no.

Follow the on-screen instructions to complete the configuration.

For more information, see "Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server" on page 49.

NOTE: After you install OpenText eDirectory, we recommend that you exclude the DIB directory on your OpenText eDirectory server from any antivirus or backup software processes. Use the OpenText eDirectory Backup Tool to back up your DIB directory.

For more information about backing up OpenText eDirectory, see "Backing Up and Restoring OpenText eDirectory," in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Non-Root User Configuring OpenText eDirectory with Root Install

There are several security threats associated when a non-root user installs and configures OpenText eDirectory using the installation tarball. The non-root user that configures OpenText eDirectory will own the binaries which will leave it vulnerable to external attacks. As a security measure, we recommend you to run the `ndsd` service as a non-root user, after installing OpenText eDirectory using the installer script as root user. The non-root user must be a non-privileged user and should not own/have any write permissions to the OpenText eDirectory binaries placed during the root installation reducing the chances of security threats. This also enables you to use the `nds-install` utility to maintain or upgrade the OpenText eDirectory server:

1. As a root user, use the `nds-install` utility to install OpenText eDirectory components on Linux systems:

   ```
   ./nds-install
   ```

   Accept the license agreement.

   For more information, see "Using the nds-install Utility to Install OpenText eDirectory Components" on page 41.

2. Export the paths as follows:

   - **Manually export the environment variables by entering the following commands:**

     ```
     export LD_LIBRARY_PATH=/opt/novell/eDirectory/lib64:/opt/novell/
     eDirectory/lib64/nds-modules:/opt/novell/lib64:$LD_LIBRARY_PATH
     ```

     ```
     export PATH=/opt/novell/eDirectory/bin:/opt/novell/eDirectory/
     sbin:/opt/novell/eDirectory/bin:$PATH
     ```

     ```
     export MANPATH=/opt/novell/man:/opt/novell/eDirectory/man:$MANPATH
     ```

     ```
     export TEXTDOMAINDIR=/opt/novell/eDirectory/share/
     locale:$TEXTDOMAINDIR
     ```

     Use the ndspath script to export the environment variables by performing the following steps:

     If you do not want to export the paths manually, prefix the ndspath script to the utility.

     - Run the utility you want as follows:

       ```
       /opt/novell/eDirectory/bin/ndspath utility_name_with_parameters
       ```

     - Export the paths in the current shell as follows:

       ```
       . /opt/novell/eDirectory/bin/ndspath
       ```

       After entering the above commands, run the utilities as you would normally do.

     - Call the script in your profile, `bashrc`, or similar scripts. Therefore, whenever you log in or open a new shell, you can start using the utilities directly.

3 As a non-root user, use the `ndsconfig` or `ndsmanage` utility to configure the eDirectory server after performing root installation of OpenText eDirectory. A sample command has been shown below:

```
ndsconfig new [-t <treename>] [-n <server_context>] [-a <admin_FDN>] [-w <admin password>] [-i] [-S <server_name>] [-d <path_for_dib>] [-m <module>] [e] [-L <ldap_port>] [-l <SSL_port>] [-o <http_port>] -O <https_port>] [-p <IP address:[port]>] [-c] [-b <port_to_bind>] [-B <interface1@port1>, <interface2@port2>,..] [-D <custom_location>] [--config-file <configuration_file>] [--configure-eba-now <yes/no>]
```

For example:

```
ndsconfig new -t mary-tree -n novell -a admin.novell -S linux1 -d /home/mary/inst1/data -b 1025 -L 1026 -l 1027 -o 1028 -O 1029 -D /home/mary/inst1/var --config-file /home/mary/inst1/nds.conf --configure-eba-now yes
```

NOTE: ◆OpenText eDirectory cannot be configured using the standard ports. The port range should be between 1024 - 65535.

- The non-root user should have the required permissions to the location where the OpenText eDirectory configuration file is stored unless the configuration will fail.

To start `systemd` specific to a user, login to the machine where OpenText eDirectory is installed as root user and run the following command:

```
systemctl start user@<uid>.service
```

where `uid` is the User ID of the user.

For example, `systemctl start user@1001.service`

To enable persistent `systemd` user instance, run the following command:

```
loginctl enable-linger user
```

Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server

After installing OpenText eDirectory, configure the OpenText eDirectory replica server using the ndsconfig utility. You must have Administrator rights to use the ndsconfig utility. When this utility is used with arguments, it validates all arguments and prompts for the password of the user having Administrator rights. If the utility is used without arguments, ndsconfig displays a description of the utility and available options. This utility can also be used to remove the OpenText eDirectory Replica Server and change the current configuration of OpenText eDirectory Server. For more information, see "The ndsconfig Utility" on page 157.

NOTE: From OpenText eDirectory 9.3.0 onwards, if it is 9.3.0 master server then adding any lower versions of OpenText edirectory as replica is not supported, with or without EBA.

Prerequisite for Configuring OpenText eDirectory in a Specific Locale

If you want to configure OpenText eDirectory in a specific locale, you need to export LC_ALL and LANG to that particular locale before OpenText eDirectory configuration. For example, to configure OpenText eDirectory in the Japanese locale, enter the following:

```
export LC_ALL=ja
```

```
export LANG=ja
```

Creating A New Tree

Use the following syntax:

```
ndsconfig new [-m <modulename>] [-i] [-S <server name>] [-t <tree_name>] [-
n <server context>] [-d <path_for_dib>][-P <LDAP URL(s)>] [-L <ldap_port>]
[-l <ssl_port>] [-o http port] [-O https port] [-e] -a <admin FDN> [-R] [-
c] [-w <admin password>] [-b <port to bind>] [-B <interface1@port1,
interface2@port2,..>] [-D <path_for_data>] [--config-file <configuration
file>] [--configure-eba-now <yes/no>] [--pki-default-rsa-keysize <2048/
4096/8192>] [--pki-default-ec-curve <P256/P384/P521>] [--pki-default-cert-
life <in years>]
```

A new tree is installed with the specified tree name and context.

There is a limitation on the number of characters in the *tree_name*, *admin FDN* and *server FDN* variables. The maximum number of characters allowed for these variables is as follows:

- *tree_name*: 32 characters
- *admin FDN*: 255 characters
- *server FDN*: 255 characters

IMPORTANT: Though OpenText eDirectory allows you to set the NCP server object's FDN up to 256 characters, we recommend that you restrict the variable to a much lesser value because OpenText eDirectory creates other objects of greater length based on the length of this object.

If the parameters are not specified in the command line, ndsconfig prompts you to enter values for each of the missing parameters.

Or, you can also use the following syntax:

```
ndsconfig def [-t <treename>] [-n <server context>] [-a <admin FDN>] [-w
<admin password>] [-c] [-i] [-S <server name>] [-d <path for dib>] [-m
<module>] [-e] [-L <ldap port>] [-l <SSL port>] [-o <http port>] [-O <https
port>] [-D <custom_location>] [--config-file <configuration_file>] [--
configure-eba-now <yes/no>]
```

A new tree is installed with the specified tree name and context. If the parameters are not specified in the command line, ndsconfig takes the default value for each of the missing parameters.

For example, to create a new tree, you could enter the following command:

```
ndsconfig new -t corp-tree -n o=company -a cn=admin.o=company
```

NOTE: A new option called `--enable-pbkdf2` has been added to the `ndsconfig` command in OpenText eDirectory 9.2 while creating a new tree. If this option is set, a password policy is created and assigned automatically to the whole tree. This password policy enables synchronization of NDS passwords with PBKDF2 passwords for all users in the tree. For more information, see Understanding Non-Reversible Password Storage in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.

Specifying Default Parameters for Default Server Certificates

OpenText eDirectory provides the option to specify the default RSA key size, Elliptic Curve and certificate life for the CA certificates and default server certificates while configuring a new OpenText eDirectory tree. You can use the following commands to specify default parameters for the CA and default server certificates while configuring a new OpenText eDirectory tree using `ndsconfig new`:

- **pki-default-rsa-keysize:** To specify the key size for RSA certificates. Allowed values are 2048, 4096 and 8192 bits.
- **pki-default-ec-curve:** To specify the curve limit for EC certificates. Allowed values are P256, P384 and P521.
- **pki-default-cert-life:** To specify the certificate life in number of years.

These attributes can be set in the `ndsconfig new` while installing a new OpenText eDirectory server.

The values specified here will be set on corresponding attributes on the Organizational CA object when the new tree is configured.

For more information, see Creating an Organizational Certificate Authority Object in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Adding a Server into an Existing Tree

Use the following syntax:

```
ndsconfig add [-t <treename>] [-n <server context>] [-a <admin FDN>] [-w
<admin password>] [-e] [-P <LDAP URL(s)>][-L <ldap port>] [-l <SSL port>]
[-o <http port>] [-O <https port>] [-S <server name>] [-d <path for dib>]
[-m <module>] [-p <IP address:[port]>] [-R] [-c] [-b <port to bind>] [-B
<interface1@port1>, <interface2@port2>,..] [-D <custom_location>] [--
config-file <configuration_file>] [-E] [--configure-eba-now <yes/no>]
```

OpenText eDirectory adds a server to an existing tree in the specified context. If the context that the user wants to add the server object to does not exist, ndsconfig creates the context and adds the server.

To enable Enhanced Background Authentication (EBA), specify *yes* to the `--configure-eba-now` switch in the ndsconfig upgrade command. Otherwise, specify *no* to configure it later. If you do not pass the `--configure-eba-now` switch to the ndsconfig command, OpenText eDirectory prompts you to specify your choice. By default, the configuration is set to *no*.

To add a secondary EBA-enabled server to the tree, you must have an EBA CA configured in the tree. If the EBA CA is not present, first add the server without enabling EBA and then upgrade the server to host the EBA CA. Otherwise, the configuration of the secondary server fails.

LDAP and security services can also be added after eDirectory has been installed into the existing tree.

For example, to add a server into an existing tree, you could enter the following command:

```
ndsconfig add -t corp-tree -n o=company -a cn=admin.o=company -S srv1
```

You can enable encrypted replication in the server you want to add using the `-E` option. For more information on encrypted replication, see "Encrypted Replication" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

NOTE: We have deprecated the support for Encrypted Replication in OpenText eDirectory 9.2.7 release.

Removing a Server Object And Directory Services From a Tree

Use the following syntax:

```
ndsconfig rm [-a <admin FDN>] [-w <admin password>] [-p <IP address:[port]>] [-c]
```

eDirectory and its database are removed from the server.

NOTE: The HTML files created using iMonitor will not be removed. You must manually remove these files from `/var/opt/novell/eDirectory/data/dsreports` before removing OpenText eDirectory.

For example, to remove the OpenText eDirectory Server object and directory services from a tree, you could enter the following command:

```
ndsconfig rm -a cn=admin.o=company
```

ndsconfig Utility Parameters

ndsconfig Parameter	Description
`new`	Creates a new OpenText eDirectory tree. If the parameters are not specified in the command line, ndsconfig prompts you to enter values for each of the missing parameters.
`def`	Creates a new OpenText eDirectory tree. If the parameters are not specified in the command line, ndsconfig takes the default value for each of the missing parameters.
`add`	Adds a server into an existing tree. Also adds LDAP and SAS services, after OpenText eDirectory has been configured in the existing tree.
`rm`	Removes the Server object and directory services from a tree. **NOTE:** This option does not remove the key material objects. These objects must be removed manually.
`upgrade`	Upgrades OpenText eDirectory to a later version.

ndsconfig Parameter	Description
`-i`	While configuring a new tree, ignores checking whether a tree of the same name exists. Multiple trees of the same name can exist.
`-S` *`server name`*	Specifies the server name. The server name can also contain dots (for example, netiq.com). Because ndsconfig is a command line utility, using containers with dotted names requires that those dots be escaped out, and the parameters containing these contexts must be enclosed in double quotes. For example, to install a new OpenText eDirectory tree on a Linux server using `netiq.com` as the name of the O, use the following command: `ndsconfig new -a "admin.novell\\.com" -t netiq_tree -n "OU=servers.O=netiq\\.com"` The Admin name and context and the server context parameters are enclosed in double quotes, and only the '.' in `netiq.com` is escaped using the '\\' (backslash) character. You can also use this format when installing a server into an existing tree. **NOTE:** You cannot start a name with a dot. For example, you cannot install a server that has the name ".novell", because it starts with a dot ('.').
`-t` *`treename`*	The tree name to which the server has to be added. It can have a maximum of 32 characters. If not specified, ndsconfig takes the tree name from the `n4u.nds.tree-name` parameter that is specified in the `/etc/opt/novell/eDirectory/conf/nds.conf` file. The default treename is `$LOGNAME-$HOSTNAME-NDStree`.
`-n` *`server context`*	Specifies the context of the server in which the server object is added. It can have a maximum of 64 characters. If the context is not specified, ndsconfig takes the context from the configuration parameter `n4u.nds.server-context` specified in the `/etc/opt/novell/eDirectory/conf/nds.conf` file. The server context should be specified in the typed form. The default context is `org`.
`-d` *`path for dib`*	The directory path where the database files will be stored.
`-r`	This option forcefully adds the replica of the server regardless of the number of servers already added to the server.
`-L` *`ldap_port`*	Specifies the TCP port number on the LDAP server. If the default port 389 is already in use, it prompts for a new port.
`-l` *`ssl_port`*	Specifies the SSL port number on the LDAP server. If the default port 636 is already in use, it prompts for a new port.
`-a` *`admin FDN`*	Specifies the fully distinguished name of the User object with Supervisor rights to the context in which the server object and Directory services are to be created. The admin name should be specified in the typed form. It can have a maximum of 64 characters. The default admin name is admin.org.
`-e`	Enables clear text passwords for LDAP objects.

ndsconfig Parameter	Description
`-m modulename`	Specifies the module name to configure. While configuring a new tree, you can configure only the ds module. After configuring the ds module, you can add the NMAS, LDAP, SAS, SNMP, HTTP services, and OpenText SecretStore (ss) using the `add` command. If the module name is not specified, all the modules are installed. **NOTE:** If you do not want to configure the SecretStore during OpenText eDirectory upgrade through `nds-install`, pass the `no_ss` value to this option. For example, `nds-install '-m no_ss'`.
`-o`	Specifies the HTTP clear port number.
`-O`	Specifies the HTTP secure port number.
`-p <IP address:[port]>`	This option is used for secondary server addition (`add` command) to a tree. It specifies the IP address of the remote host that holds a replica of the partition to which this server is being added. The default port number is 524. This helps in faster lookup of the tree since it avoids SLP lookup.
`-R`	By default a replica of the partition to which the server is added would be replicated to the local server. This option disallows adding replicas to the local server.
`-c`	This option avoids prompts during ndsconfig operation, such as yes/no to continue the operation, or prompt to re-enter port numbers when there is a conflict, etc. The user receives prompts only for entering mandatory parameters if they are not passed on command line.
`-w <admin password>`	This option allows passing the admin user password in clear text. **NOTE:** Since password is passed in clear text, this is not recommended as a safe option owing to password insecurity.
`-E`	Enables encrypted replication for the server you are trying to add.
`-j`	Jumps or overrides the health check option before installing OpenText eDirectory.
`-b port to bind`	Sets the default port number on which a particular instance should listen on. This sets the default port number on `n4u.server.tcp-port` and `n4u.server.udp-port`. If an NCP port is passed using the `-b` option, then it is assumed to be the default port and the TCP and UDP parameters are updated accordingly. **NOTE:** `-b` and `-B` are exclusively used.

`-B interface1@port1, interface2@port2,...`	Specifies the port number along with the IP address or interface. For example:
	
	`-B eth0@524`
	
	or
	
	`-B 100.1.1.2@524`
	
	NOTE: `-b` and `-B` are mutually exclusive. If the administrator specifies only `@<port-number>` for this option, the NCP, http and https services will bind to all the available interfaces.

ndsconfig Parameter	Description
`--config-file` *`configuration file`*	Specify the absolute path and file name to store the `nds.conf` configuration file. For example, to store the configuration file in the `/etc/opt/novell/eDirectory/ directory`, enter `--config-file /etc/opt/novell/eDirectory/nds.conf`.
`-P` *<LDAP URL(s)>*	Allows the LDAP URLs to configure the LDAP interface on the LDAP Server object. For example: `-P ldap://1.2.3.4:1389,ldaps://1.2.3.4:1636`
`-D` *`path_for_data`*	Creates the `data`, `dib`, and `log` directories in the path mentioned.
`set valuelist`	Sets the value for the specified OpenText eDirectory configurable parameters. It is used to set the bootstrapping parameters before configuring a tree. When configuration parameters are changed, ndsd needs to be restarted for the new value to take effect. However, for some configuration parameters, ndsd need not be restarted. These parameters are listed below: ◆ `n4u.nds.inactivity-synchronization-interval` ◆ `n4u.nds.synchronization-restrictions` ◆ `n4u.nds.janitor-interval` ◆ `n4u.nds.backlink-interval` ◆ `n4u.nds.drl-interval` ◆ `n4u.nds.flatcleaning-interval` ◆ `n4u.nds.server-state-up-threshold` ◆ `n4u.nds.heartbeat-schema` ◆ `n4u.nds.heartbeat-data` ◆ `n4u.server.fips_tls` ◆ `n4u.server.eba_enabled`
`get help paramlist`	Use to view the help strings for the specified OpenText eDirectory configurable parameters. If the parameter list is not specified, ndsconfig lists the help strings for all the OpenText eDirectory configurable parameters.
`set valuelist`	Sets the value for the specified OpenText eDirectory configurable parameters. It is used to set the bootstrapping parameters before configuring a tree. When configuration parameters are changed, ndsd needs to be restarted for the new value to take effect.
`get paramlist`	Use to view the current value of the specified OpenText eDirectory configurable parameters. If the parameter list is not specified, ndsconfig lists all the OpenText eDirectory configurable parameters.
`configure -eba-now`	Use this switch to configure your OpenText eDirectory server for enhanced background authentication.

Using ndsconfig to Configure Multiple Instances of OpenText eDirectory 9.3

You can configure multiple instances of OpenText eDirectory 9.3 on a single host. With the multiple instances feature support in OpenText eDirectory 9.3, you can configure the following:

- Multiple instances of OpenText eDirectory on a single host
- Multiple trees for different users on a single host
- Multiple replicas of the same tree or partition on a single host

---

WARNING: Configuring multiple trees for the same user is not supported. OpenText does not support instances of servers in different trees for a user. If you want to configure servers in multiple trees, use different user accounts.

The following table lists the platforms that support the multiple instances:

Feature	Linux	Windows
Multiple instances support	✓	✗

The method to configure multiple instance is similar to configuring a single instance multiple times. Each instance should have unique instance identifiers, such as the following:

- Different data and log file location

  You can use the ndsconfig `--config-file`, `-d`, and `-D` options to do this.
- Unique port number for the instance to listen to

  You can use the ndsconfig `-b` and `-B` options to do this.
- Unique server name for the instance

  You can use the ndsconfig `-S` *server name* option to do this.

  IMPORTANT: During OpenText eDirectory configuration, the default NCP server name is set as the host server name. When configuring multiple instances, you must change NCP server name. Use the `ndsconfig` command line option, `-S <server_name>` to specify a different server name.

  When configuring multiple instances, either on the same tree or on different trees, the NCP server name should be unique.

Need for Multiple Instances

Multiple instances arose from the need to:

- Leverage high-end hardware by configuring more than one instance of OpenText eDirectory.
- Pilot your setup on a single host before investing on the required hardware.

Sample Scenarios for Deploying Multiple Instances

Multiple instances that belong to the same or multiple trees can be used in the following scenarios effectively.

eDirectory in a Large Enterprise

- In large enterprises, you can provide load balancing and high availability of OpenText eDirectory services.

  For example, if you have three replica servers running LDAP services on ports 1524, 2524, and 3524, respectively, you can configure a new instance of OpenText eDirectory and provide a high-availability LDAP service on a new port 636.
- You can leverage high-end hardware across departments in an organization by configuring multiple instances on a single host.

OpenText eDirectory in an Evaluation Setup

- **Universities:** Many enthusiasts (students) can evaluate OpenText eDirectory from the same host using the multiple instances.
- **Training for eOpenText Directory administration:**
  - Participants can try out administration using the multiple instances.
  - Instructors can use a single host to teach a class of students. Each student can have his own tree.

Using Multiple Instances

OpenText eDirectory 9.3 makes it very easy for you to configure multiple instances. To effectively use multiple instances, you need to plan the setup and then configure the multiple instances.

- “Planning the Setup” on page 61
- “Configuring Multiple Instances” on page 58

Planning the Setup

To use this feature effectively, we recommend that you plan the OpenText eDirectory instances and ensure that each instance has definite instance identifiers like the hostname, port number, server name, or the configuration file.

While configuring multiple instances, you need to ensure that you have planned for the following:

- Location of the configuration file
- Location of the variable data (like log files)
- Location of the DIB
- NCP interface, unique identifying port for every instance, and ports of other services (like LDAP, LDAPS, HTTP, and HTTP secure port)
- Unique server name for every instance

Configuring Multiple Instances

You can configure multiple instances of OpenText eDirectory using the ndsconfig utility. The following table lists the ndsconfig options you need to include when configuring multiple instances.

NOTE: All the instances share the same server key (NICI).

Option	Description
--config-file	Specifies the absolute path and filename to store the `nds.conf` configuration file. For example, to store the configuration file in the `/etc/opt/novell/eDirectory/` directory, use `--config-file /etc/opt/novell/eDirectory/nds.conf`.
-b	Specifies the port number where the new instance should listen. **NOTE:** `-b` and `-B` are exclusively used.

-B	Specifies the port number along with the IP address or interface. For example:
	
	`-B eth0@524`
	
	or
	
	`-B 100.1.1.2@524`
	
	NOTE: `-b` and `-B` are exclusively used.

-D	Creates the `data`, `dib`, and `log` directories in the path specified for the new instance.
S	Specifies the server name.

Using the above-mentioned options, you can configure a new instance of OpenText eDirectory.

You can also configure a new instance using the ndsmanage utility. For more information, refer to “Creating an Instance through ndsmanage” on page 59.

Managing Multiple Instances

This section includes the following information:

- “The ndsmanage Utility” on page 58
- “Identifying a Specific Instance” on page 61
- “Invoking a Utility for a Specific Instance” on page 61

The ndsmanage Utility

The ndsmanage utility enables you to do the following:

- List the instances configured
- Create a new instance

- Do the following for a selected instance:
  - List the replicas on the server
  - Start the instance
  - Stop the instance
  - Run DSTrace (ndstrace) for the instance
  - Deconfigure the instance
- Start and Stop all instances

Listing the Instances

The following table describes how to list the eDirectory instances.

Table 3-1 *ndsmanage Usage for Listing the Instances*

Syntax	Description
ndsmanage	Lists all the instances configured by you.
ndsmanage -a|--all	List instances of all the users who are using a particular installation of eDirectory.
ndsmanage *username*	List the instances configured by a specific user

The following fields are displayed for every instance:

- Configuration file path
- Server FDN and port
- Status (whether the instance is active or inactive)

NOTE: This utility lists all the instances configured for a single binary.

Creating an Instance through ndsmanage

To create a new instance through ndsmanage:

1. Enter the following command:

   ```
   ndsmanage
   ```

2. Enter c to create a new instance.

   You can either create a new tree or add a server to an existing tree. Follow the instructions on the screen to create a new instance.

Performing Operations for a Specific Instance

You can perform the following operations for every instance:

Other than the ones listed below, you can also run DSTrace for a selected instance.

Starting a Specific Instance

To start an instance configured by you, do the following:

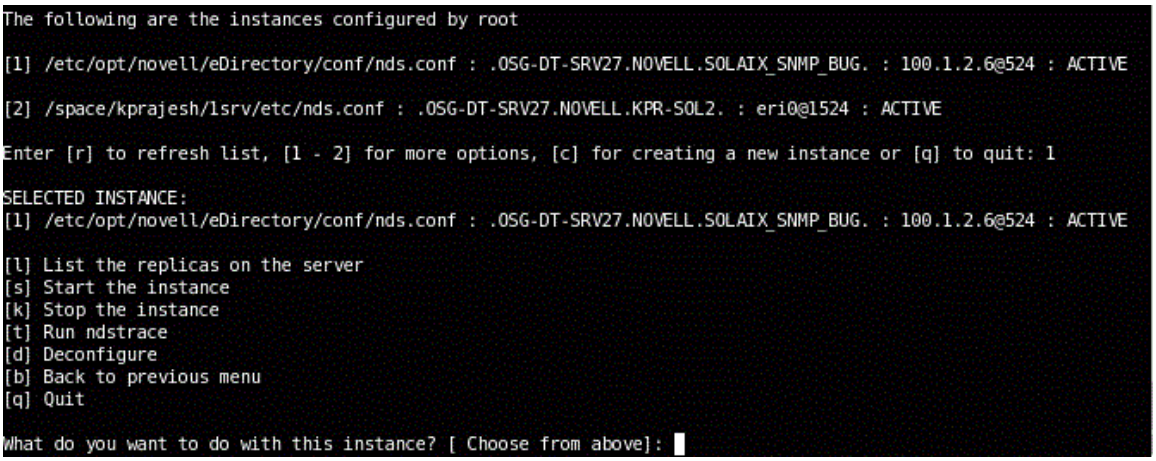
1. Enter the following:

   ```
   ndsmanage
   ```

2. Select the instance you want to start.

   The menu expands to include the options you can perform on a specific instance.

Figure 3-1 *ndsmanage Utility Output Screen with Instance Options*



3. Enter s to start the instance.

Alternatively, you can also enter the following at the command prompt:

```
ndsmanage start --config-file
configuration_file_of_the_instance_configured_by_you
```

Stopping a Specific Instance

To stop an instance configured by you, do the following:

1. Enter the following:

   ```
   ndsmanage
   ```

2. Select the instance you want to stop.

   The menu expands to include the options you can perform on a specific instance. For more information, refer to ndsmanage Utility Output Screen with Instance Options (page 60).

3. Enter k to stop the instance.

Alternatively, you can also enter the following at the command prompt:

```
ndsmanage stop --config-file
configuration_file_of_the_instance_configured_by_you
```

Deconfiguring an Instance

To deconfigure an instance, do the following:

1. Enter the following:

`ndsmanage`

2. Select the instance you want to deconfigure.

   The menu expands to include the options you can perform on a specific instance. For more information, refer to ndsmanage Utility Output Screen with Instance Options (page 60).

3. Enter `d` to deconfigure the instance.

Starting and Stopping All Instances

You can start and stop all the instances configured by you.

Starting all the Instances

To start all the instances configured by you, enter the following at the command prompt:

```
ndsmanage startall
```

To start a specific instance, refer to "Starting a Specific Instance" on page 59.

Identifying a Specific Instance

While configuring multiple instances, you assign a hostname, port number, and a unique configuration file path to every instance. This hostname and port number are the instance identifiers.

Most of the utilities have the `-h` *`hostname`*`:`*`port`* or `--config-file` *`configuration_file_location`* option that enables you to specify a particular instance. See the man pages of the utilities for more information.

Invoking a Utility for a Specific Instance

If you want to run a utility for a specific instance, you need to include the instance identifier in the utility command. The instance identifiers are the path of the configuration file, and the hostname and port number. You can use the `--config-file` *`configuration_file_location`* or the `-h` *`hostname`*`:`*`port`* to do so.

If you do not include the instance identifiers in the command, the utility displays the various instances you own and prompts you to select the instance you want to run the utility for.

For example, to run DSTrace for a specific utility using the `--config-file` option, you would enter the following:

```
ndstrace --config-file configuration_filename_with_location
```

Sample Scenario for Multiple Instances

Mary is a non-root user who wants to configure two trees on a single host machine for a single binary.

Planning the Setup

Mary specifies the following instance identifiers.

- **Instance 1:**

Port number the instance should listen on	1524
Configuration file path	`/home/maryinst1/nds.conf`
DIB directory	`/home/mary/inst1/var`

- **Instance 2:**

Port number the instance should listen on	2524
Configuration file path	`/home/mary/inst2/nds.conf`
DIB directory	`/home/mary/inst2/var`

Configuring the Instances

To configure the instances based on the above mentioned instance identifiers, Mary must enter the following commands.

- **Instance 1:**

```
ndsconfig new -t mytree -n o=novell -a cn=admin.o=company -b 1524 -D
/home/mary/inst1/var --config-file /home/mary/inst1/nds.conf
```

- **Instance 2:**

```
ndsconfig new -t corptree -n o=novell -a cn=admin.o=company -b 2524 -D
/home/mary/inst2/var --config-file /home/mary/inst2/nds.conf
```

Invoking a Utility for an Instance

If Mary wants to run the DSTrace utility for instance 1 that is listening on port 1524, with its configuration file in `/home/mary/inst1/nds.conf` location and its DIB file located in `/home/mary/inst1/var`, then she can run the utility as follows:

```
ndstrace --config-file /home/mary/inst1/nds.conf
```

or

```
ndstrace -h 164.99.146.109:1524
```

If Mary does not specify the instance identifiers, the utility displays all the instances owned by Mary and prompts her to select an instance.

Listing the Instances

If Mary wants to know details about the instances in the host, she can run the ndsmanage utility.

- To display all instances owned by Mary:

```
ndsmanage
```

- To display all instances owned by John (username is john):

```
ndsmanage john
```

- To display all instances of all users that are using a particular installation of OpenText eDirectory:

  ```
  ndsmanage -a
  ```

Using ndsconfig to Install a Linux Server into a Tree with Dotted Name Containers

You can use ndsconfig to install a Linux server into an OpenText eDirectory tree that has containers using dotted names (for example, novell.com).

Because ndsconfig is a command line utility, using containers with dotted names requires that those dots be escaped out, and the parameters containing these contexts must be enclosed in double quotes. For example, to install a new eDirectory tree on a Linux server using “O=netiq.com” as the name of the O, use the following command:

```
ndsconfig new -a 'admin.netiq.com' -t netiq_tree -n
'OU=servers.O=netiq.com'
```

The Admin name and context and the server context parameters are enclosed in double quotes, and only the dot (’.’) in novell.com is escaped using the ’\’ (backslash) character.

You can also use this format when installing a server into an existing tree.

NOTE: You should use this format when entering dotted admin name and context while using utilities such as DSRepair, Backup, DSMerge, DSLogin, and ldapconfig.

Using the nmasinst Utility to Configure NMAS

By default, ndsconfig configures NMAS. You can also use nmasinst to configure NMAS.

ndsconfig only configures NMAS and does not install the login methods. To install these login methods, you can use nmasinst.

IMPORTANT: You must configure OpenText eDirectory with ndsconfig before you install the NMAS login methods. You must also have administrative rights to the tree.

- “Configuring NMAS” on page 63
- “Installing Login Methods” on page 64

Configuring NMAS

By default, ndsconfig configures NMAS. You can also use nmasinst for the same.

To configure NMAS and create NMAS objects in eDirectory, enter the following at the server console command line:

```
nmasinst -i admin.context tree_name
```

nmasinst prompts you for a password.

This command creates the objects in the Security container that NMAS needs, and installs the LDAP extensions for NMAS on the LDAP Server object in OpenText eDirectory.

The first time NMAS is installed in a tree, it must be installed by a user with enough rights to create objects in the Security container. However, subsequent installs can be done by container administrators with read-only rights to the Security container. nmasinst will verify that the NMAS objects exist in the Security container before it tries to create them.

nmasinst does not extend the schema. The NMAS schema is installed as part of the base OpenText eDirectory schema.

Installing Login Methods

To install login methods using nmasinst, enter the following at the server console command line:

```
nmasinst -addmethod admin.context tree_name config.txt_path
```

The last parameter specifies the `config.txt` file for the login method that is to be installed. A `config.txt` file is provided with each login method.

Here is an example of the `-addmethod` command:

```
nmasinst -addmethod admin.netiq MY_TREE ./nmas-methods/novell/Simple
Password/config.txt
```

If the login method already exists, nmasinst will update it.

For more information, see Managing Login and Post-Login Methods and Sequences in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.

Non-root User SNMP Configuration

`NICI` and `NOVLsubag` should be installed as root user.

1. Root User Installing `NICI`. Refer to "Root User Installing NICI" on page 45
2. Root User Installing `NOVLsubag`.

   To install `NOVLsubag`, complete the following procedure:

   Enter the following command:

   ```
   rpm -ivh --nodeps NOVLsubag_rpm_file_name_with_path
   ```

   For example:

   ```
   rpm -ivh --nodeps novell-novell-NOVLsubag-9.2.0-0.x86_64.rpm
   ```

3. Export the paths as follows:

   Manually export the environment variables.

   ```
   export LD_LIBRARY_PATH=custom_location/opt/novell/eDirectory/lib64:/
   opt/novell/eDirectory/lib64/nds-modules:/opt/novell/
   lib64:$LD_LIBRARY_PATH
   ```

   ```
   export   PATH=/opt/novell/eDirectory/bin:$PATH
   ```

   ```
   export MANPATH=/opt/novell/man:$MANPATH
   ```

Locating Log Files

ndsd.log

The `ndsd.log` file contains information about OpenText eDirectory server-related messages, such as, server shutdown and start messages and PKI and LDAP services start and shutdown messages. By default, this file is located in the `/var/opt/novell/eDirectory/log` directory.

You can increase the debug level for the `ndsd.log` file by modifying the following variable in the `nds.conf` file from `/etc/opt/novell/eDirectory/conf/nds.conf` file.

```
n4u.server.log-levels=Logxxxx
```

For more information on ndsd log levels, see Managing Error Logging in OpenText eDirectory.

Specifying the Log File Size on Linux

To specify the size of the log file, use the `n4u.server.log-file-size` parameter in the `nds.conf` file. The maximum file limit can be 2 GB and the default file size is 1 MB. However, you can set the file size to below 1 MB also.

This setting is not applicable to the `ndsd.log` file.

If the log file size reaches the specified limit, then logger overwrites the log file from the start.

4 Installing or Upgrading OpenText eDirectory CE 24.4 (v9.3) on Windows

Use the following information to install or upgrade OpenText eDirectory 9.3 on a Windows platform:

- "System Requirements" on page 67
- "Prerequisites" on page 68
- "Hardware Requirements" on page 70
- "Forcing the Backlink Process to Run" on page 71
- "Installing OpenText eDirectory on Windows" on page 71
- "Upgrading OpenText eDirectory on Windows" on page 83

IMPORTANT: OpenText eDirectory 9.3 lets you install OpenText eDirectory for Windows without the Novell Client. For more information, see "Installing or Updating OpenText eDirectory 9.3 on a Windows Server" on page 72.

System Requirements

You must install OpenText eDirectory on one of the following platforms:

- Windows Server 2016 at a minimum and Windows Server 2022

IMPORTANT: Windows desktop versions are not supported.

OpenText eDirectory also requires the following:

For the most recent information about the system requirements, see the Release Notes.

- An assigned IP address
- Administrative rights to the Windows server and to all portions of the OpenText eDirectory tree that contain domain-enabled User objects. For an installation into an existing tree, you need administrative rights to the Tree object so that you can extend the schema and create objects.

Refer to the OS recommended hardware requirements for your Windows server.

Prerequisites

IMPORTANT: Check the currently installed OpenText and Third Party applications to determine if OpenText eDirectory 9.3 is supported before upgrading your existing eDirectory environment. You can find out the current status for OpenText products in the TID 7003446 (http://www.novell.com/support/kb/doc.php?id=7003446) It is also highly recommended to back up OpenText eDirectory prior to any upgrades.

- ❒ Because NTFS provides a safer transaction process than a FAT file system provides, you can install OpenText eDirectory only on an NTFS partition. Therefore, if you have only FAT file systems, do one of the following:
  - Create a new partition and format it as NTFS.

    Use Disk Administrator. Refer to the Windows Server documentation for more information.
  - Convert an existing FAT file system to NTFS, using the CONVERT command.

    Refer to the Windows Server documentation for more information.

  If your server only has a FAT file system and you forget or overlook this process, the installation program prompts you to provide an NTFS partition.
- ❒ (Conditional) NICI 3.3 and OpenText eDirectory 9.3 support key sizes up to 8192 bits for RSA encryption. If you want to use a 8K key size, every server must be upgraded to OpenText eDirectory 9.3. In addition, every workstation using the management utilities, for example, Identity Console, must have NICI 3.3 installed on it.

  When you upgrade your Certificate Authority (CA) server to OpenText eDirectory 9.3, the key size will not change but will still be 2K. The only way to create a 8K key size is recreate the CA on an OpenText eDirectory 9.3 server. In addition, you would have to change the default from 2K to 8K for the key size, during the CA creation.

  NOTE: The Windows Silent installer requires NICI 3.3 installed on the system.
- ❒ Ensure to obtain 168-bit 3DES tree key for your OpenText eDirectory servers.
- ❒ If you are upgrading to OpenText eDirectory 9.3, make sure you have the latest OpenText eDirectory patches installed on all non OpenText eDirectory 9.3 servers in the tree. You can get OpenText eDirectory patches from the OpenText Support (http://support.novell.com) Web site.
- ❒ .NET Management Framework 4.0 or above is required.
- ❒ Make sure you have the latest Windows Server Service Packs installed. The latest updated Windows Server Service Pack needs to be installed after the installation of the Windows SNMP service.
- ❒ If you are upgrading from a previous version of OpenText eDirectory, it must be eDirectory 9.3.0 or later. For more information on determining the eDirectory version, see “Determining the version of OpenText eDirectory” on page 70.
- ❒ (Conditional) If you are installing a secondary server into an existing tree as a non-administrator OpenText eDirectory user, ensure that you have the following rights:
  - Supervisor rights to the container the server is being installed into.
  - Supervisor rights to the partition where you want to add the server.

    NOTE: This is required for adding the replica when the replica count is less than 3.

- All Attributes rights: read, compare, and write rights over the W0.KAP.Security object and W1.KAP.Security object.
- Entry rights: browse rights over Security container object.
- All Attributes rights: read and compare rights over Security container object.
- (Conditional) If the W1.KAP.Security object exists, all attributes rights: read, compare, and write rights over this object. For more information about the W1.KAP.Security object, see Creating an AES 256-Bit Tree Key in the *NICI Administration Guide*.

❒ (Conditional) If you are installing a secondary server into an existing tree as a non-administrator user, ensure that at least one of the servers in the tree has the same or higher OpenText eDirectory version as that of the secondary being added as container admin. In case the secondary being added is of later version, then the schema needs to be extended by the admin of the tree before adding the secondary using container admin.

❒ While configuring eDirectory, you must enable SLP services and an NCP port (the default is 524) in the firewall to allow the secondary server addition. The NCP port must be configured to allow both inbound and outbound traffic.

Additionally, you can enable the following service ports, based on your requirements:

- LDAP clear text - 389
- LDAP secured - 636
- HTTP clear text - 8028
- HTTP secured - 8030

If you have enabled user-defined ports, you must specify these ports while configuring OpenText eDirectory.

❒ If you are installing OpenText eDirectory on a virtual machine having a DHCP address or on a physical or virtual machine in which SLP is not broadcast, ensure that the Directory Agent is configured in your network.

❒ If you do not have the latest Platform Agent (PA) installed while upgrading to OpenText eDirectory 9.3, please run the `Novell_Audit_PlatformAgent_Win64.exe` file from the `<C:\NetIQ\eDirectory\auditds/` location to install.

❒ The OpenText eDirectory Management Toolbox (eMBox) lets you access all of the OpenText eDirectory back-end utilities remotely, as well as on the server. The command line client is a Java application. To run it, you must install the latest version of Oracle or Azul Java (1.8 or above). You must also ensure to upgrade any older version of Java by installing the patch upgrades available. Once you have the latest version of Java installed, export any of the following environment variables:

- `EDIR_JAVA_HOME`
- `JAVA_HOME`
- `JRE_HOME`

NOTE: ◆If you are using any prior version of OpenText eDirectory 9.0 SP4, to run the command line client, you must have access to the Java Runtime Environment.

- Installing Identity Manager 4.8.7 will upgrade Java dependent packages in OpenText eDirectory from JRE 8 to JRE 11.

Determining the version of OpenText eDirectory

To determine the version of OpenText eDirectory, follow one of the steps mentioned below:

- Run iMonitor.

  On the Agent Summary page, click Known Servers. Then under Servers Known to Database, click Known Servers. The Agent Revision column displays the internal build number for each server. For example, an Agent Revision number for eDirectory 9.3 might be 40301.00.

  For information on running OpenText iMonitor, see "Accessing OpenText iMonitor" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

- Run `NDSCons.exe`.

  In the Windows Control Panel, double-click OpenText eDirectory Services. In the Services column, select ds.dlm, then click Configure. The Agent tabs displays both the marketing string (for example, OpenText eDirectory 9.3) and the internal build number (for example, 40301.00).

- View the properties of an `ds.dlm` file.

  Right-click the `.dlm` file in Windows Explorer, then click the Version tab in the Properties dialog box. This will display the version number of the utility. The default location for `ds.dlm` files is `C:\NetIQ\eDirectory`.

Configuring Static IP Address

Static IP address must be configured on the server for the OpenText eDirectory to perform efficiently. Configuring eDirectory on the servers with DHCP address can lead to unpredictable results.

Hardware Requirements

Hardware requirements depend on the specific implementation of OpenText eDirectory.

For example, a base installation of OpenText eDirectory with the standard schema requires about 74 MB of disk space for every 50,000 users. However, if you add a new set of attributes or completely fill in every existing attribute, the object size grows. These additions affect the disk space, processor, and memory needed.

Two factors increase performance: more cache memory and faster processors.

For best results, cache as much of the DIB Set as the hardware allows.

OpenText eDirectory scales well on a single processor. However, OpenText eDirectory 9.3 takes advantage of multiple processors. Adding processors improves performance in some areas—for example, logins and having multiple threads active on multiple processors. OpenText eDirectory itself is not processor intensive, but it is I/O intensive.

The following table illustrates typical system requirements for OpenText eDirectory for Windows:

Objects	Memory	Hard Disk
10,000	384 MB	144 MB
1 million	2 GB	1.5 GB
10 million	2+ GB	15 GB

Requirements for processors depend on additional services available on the computer as well as the number of authentications, reads, and writes that the computer is handling. Processes such as encryption and indexing can be processor intensive.

Forcing the Backlink Process to Run

Because the internal OpenText eDirectory identifiers change when upgrading to OpenText eDirectory, the backlink process must update backlinked objects for them to be consistent.

Backlinks keep track of external references to objects on other servers. For each external reference on a server, the backlink process ensures that the real object exists in the correct location and verifies all backlink attributes on the master of the replica. The backlink process occurs two hours after the database is open and then every 780 minutes (13 hours). The interval is configurable from 2 minutes to 10,080 minutes (7 days).

After migrating to OpenText eDirectory, we recommend that you force the backlink to run by completing the following procedure. Running the backlink process is especially important on servers that do not contain a replica.

1. Click **Start** > **Settings** > **Control Panel** > **OpenText eDirectory Services**
2. In the **Services** tab, select **ds.dlm**.
3. Click **Configure**.
4. In the **Trigger** tab, click **Backlinker**.

Installing OpenText eDirectory on Windows

This section contains the following information:

- "Installing or Updating OpenText eDirectory 9.3 on a Windows Server" on page 72
- "Server Health Checks" on page 73
- "Communicating with OpenText eDirectory through LDAP" on page 74
- "Installing NMAS Server Software" on page 75
- "Installing into a Tree with Dotted Name Containers" on page 75
- "Unattended Install and Configure to OpenText eDirectory 9.3 on Windows" on page 76
- "Locating Log Files" on page 82

Installing or Updating OpenText eDirectory 9.3 on a Windows Server

You can install OpenText eDirectory 9.3 for Windows without the Novell Client. If you install OpenText eDirectory 9.3 on a machine already containing the Novell Client, OpenText eDirectory will use the existing Client, or update it if it is not the latest version.

1. At the Windows server, log in as Administrator or as a user with administrative privileges.
2. If you have Autorun turned off, run `eDirectory_930_Windows_x86_64.exe` from the `windows` folder in the eDirectory 9.3 CD or from the downloaded file.
3. (New installations only) Select an OpenText eDirectory installation type under the **Basic** tab:
   - **Create a New eDirectory Tree** creates a new tree. Use this option if this is the first server to go into the tree or if this server requires a separate tree. The resources available on the new tree will not be available to users logged in to a different tree.
   - **Install eDirectory into an Existing Tree** incorporates this server into your eDirectory network. The server can be installed into any level of your tree.
4. Provide information in the OpenText eDirectory Installation screen:
   - If you are installing a new eDirectory server, specify a Tree name, Server object context, and Admin name and password for the new tree.

     IMPORTANT: Though OpenText eDirectory allows you to set the NCP server object's FDN up to 256 characters, OpenText recommends that you restrict the variable to a much lesser value because OpenText eDirectory creates other objects of greater length based on the length of this object.
   - If you are installing into an existing tree, specify the IP address, Tree name, Server object context, and Admin name and password of the existing tree.
   - If you are upgrading an eDirectory server, specify the Admin password.

     NOTE: OpenText eDirectory 9.3 allows you to use case sensitive passwords for all the utilities.

   For information on using dots in container names, see “Installing into a Tree with Dotted Name Containers” on page 75.
5. Specify or confirm the installation path. The default location is `C:\NetIQ\eDirectory`.
6. Specify or confirm the DIB path. The default location is `C:\NetIQ\eDirectory\DIBFiles`.
7. In **Advanced** tab, specify the following information:
   - If you want to use IPv6 addresses, select **Enable IPv6**.

     NOTE: If you do not enable IPv6 addresses during the installation process and decide to use them later, you must run the setup program again.
   - If you want to enable Enhanced Background Authentication (EBA), select **Enable EBA**.

     NOTE: If you do not enable EBA during the installation process and decide to enable it later, you must run the setup program again.

To add a secondary EBA-enabled server to the tree, you must have an EBA CA configured in the tree. If the EBA CA is not present, first add the server without enabling EBA and then upgrade the server to host the EBA CA. Otherwise, the configuration of the secondary server fails.

- Specify the **HTTP Stack ports** to use for the OpenText eDirectory administrative HTTP server.

  IMPORTANT: Make sure that the HTTP stack ports you set during the OpenText eDirectory installation are different than the HTTP stack ports you have used or will use for OpenText Identity Console. For more information, see the *OpenText Identity Console Administration Guide* (https://www.netiq.com/documentation/identity-console/identity_console-admin/data/bookinfo.html).

- Specify which **LDAP ports** to use.

  For more information, see "Communicating with OpenText eDirectory through LDAP" on page 74.

8 Click **Install**.

The installation program checks for the following components before it installs OpenText eDirectory. If a component is missing or is an incorrect version, the installation program automatically launches an installation for that component. The command prompt appears with the message - **INSTALL PASSED.** This message automatically disappears after the successful installation of NICI and the eDirectory installation continues further.

- NICI 3.3

  For more information on the Novell International Cryptographic Infrastructure (NICI), see the *NICI Administration Guide*.

9 OpenText eDirectory will install and configure all the required components automatically.

10 When the installer completes the installation, click **Finish** to exit the wizard.

IMPORTANT: Only the OpenText eDirectory administrator should be able to login to the server where OpenText eDirectory is installed.

NOTE: After you install OpenText eDirectory, we recommend you exclude the DIB directory on your eDirectory server from any antivirus or backup software processes. Use the OpenText eDirectory Backup Tool to back up your DIB directory.

For more information about backing up OpenText eDirectory, see "Backing Up and Restoring OpenText eDirectory," in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Server Health Checks

With OpenText eDirectory 9.3, when you upgrade eDirectory, a server health check is conducted by default to ensure that the server is safe for the upgrade.

- "Partitions and Replica Health" on page 189

Based on the results obtained from the health checks, the upgrade will either continue or exit as follows:

- If all the health checks are successful, the upgrade will continue.
- If there are minor errors, the upgrade will prompt you to continue or exit.
- If there are critical errors, the upgrade will exit.

See Appendix B, “OpenText eDirectory Health Checks,” on page 187 for a list of minor and critical error conditions.

Communicating with OpenText eDirectory through LDAP

When you install OpenText eDirectory, you must select a port that the LDAP server monitors so that it can service LDAP requests. The following table lists options for various installations:

Installation	Option	Result
eDirectory 9.3	Clear text (port 389)	Selects port 389.
eDirectory 9.3	Encrypted (port 636)	Selects port 636.

Port 389, the Industry-Standard LDAP Clear-Text Port

The connection through port 389 is not encrypted. All data sent on a connection made to this port is clear. Therefore, a security risk exists. For example, LDAP passwords can be viewed on a simple bind request.

An LDAP Simple Bind requires only a DN and a password. The password is in clear text. If you use port 389, the entire packet is in clear text. By default, this option is disabled during the OpenText eDirectory installation.

Because port 389 allows clear text, the LDAP server services Read and Write requests to the Directory through this port. This openness is adequate for environments of trust, where spoofing doesn't occur and no one inappropriately captures packets.

If you make a secure connection to port 636 and have a simple bind, the connection is already encrypted. No one can view passwords, data packets, or bind requests.

Port 636, the Industry-Standard Secure Port

The connection through port 636 is encrypted. TLS (formerly SSL) manages the encryption. By default, the eDirectory installation selects this port.

A connection to port 636 automatically instantiates a handshake. If the handshake fails, the connection is denied.

IMPORTANT: This default selection might cause a problem for your LDAP server. If a service already loaded on the host server (before eDirectory was installed) uses port 636, you must specify another port.

The OpenText eDirectory installation loads `nldap.nlm`, places an error message in the `dstrace.log` file, and runs without the secure port.

Scenario: Port 636 Is Already Used: Your server is running Active Directory. Active Directory is running an LDAP program, which uses port 636. You install eDirectory. The installation program detects that port 636 is already used and doesn't assign a port number for the OpenText LDAP server. The LDAP server loads and appears to run. However, because the LDAP server does not duplicate or use a port that is already open, the LDAP server does not service requests on any duplicated port.

If you are not certain that port 389 or 636 is assigned to the OpenText LDAP server, run the ICE utility. If the **Vendor Version** field does not specify OpenText, you must reconfigure LDAP Server for OpenText eDirectory and select a different port. For more information, see "Verifying That the LDAP Server Is Running" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Scenario: Active Directory Is Running: Active Directory is running. Clear-text port 389 is open. You run the `ICE` command to port 389 and ask for the vendor version. The report displays `Microsoft*`. You then reconfigure the OpenText LDAP server by selecting another port, so that the OpenText eDirectory LDAP server can service LDAP requests.

OpenText iMonitor can also report that port 389 or 636 is already open. If the LDAP server isn't working, use OpenText iMonitor to identify details. For more information, see "Verifying That the LDAP Server Is Running" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Installing NMAS Server Software

NetIQ Modular Authentication Service (NMAS) server components are installed automatically when you run the eDirectory installation program. The NDS login method is configured by default.

For more information on login methods, see Managing Login and Post-Login Methods and Sequences in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.

Installing into a Tree with Dotted Name Containers

You can install a Windows server into an OpenText eDirectory tree that has containers with dots in the names (for example, O=netiq.com or C=u.s.a). Using containers with dotted names requires that those dots be escaped with the backslash character. To escape a dot, simply put a backslash in front of any dot in a container name.

You cannot start a name with a dot. For example, you cannot create a container named ".netiq" because it starts with a dot ('.').

IMPORTANT: If your tree has containers with dotted names, you must escape those names when logging into utilities such as OpenText iMonitor, OpenText Identity Console, and DHost iConsole. For example, if your tree has "netiq.com" as the name of the O, enter *`username`*`.netiq\.com` in the **Username** field when logging in to iMonitor.

Unattended Install and Configure to OpenText eDirectory 9.3 on Windows

OpenText eDirectory 9.3 automates the OpenText eDirectory installation and upgrade so that eDirectory is installed or upgraded silently on Windows servers without human intervention.

On Windows, the unattended installation of eDirectory uses predefined text files that facilitate the unattended installation or upgrade. You can perform either of the following setup using the unattended installation of eDirectory:

- Standalone installation or upgrade of OpenText eDirectory depending on whether it is a complete installation of OpenText eDirectory or not. The standalone upgrade process upgrades only the installed files.
- Configuration of installed OpenText eDirectory. If you install OpenText eDirectory, a complete configuration of eDirectory is performed. Otherwise, when you upgrade OpenText eDirectory, the installer only configures the upgraded files.

For more information on how to mention the setup for unattended installation, refer to the section “Adding Features to the Automated Installation” on page 77.

Prerequisites

- .NET Management Framework 4.0 or above is required
- Ensure that Windows Server is updated with the latest windows patch

The following sections discuss various features that can be used to configure the unattended installation, including the install location, no display of splash screens, port configurations, additional NMAS methods, stopping and starting SNMP services, etc.

- “Response Files” on page 76
- “Adding Features to the Automated Installation” on page 77
- “Controlling Automated Installation” on page 80
- “Unattended Installation of OpenText eDirectory using Response File” on page 82

Response Files

Installing or upgrading to OpenText eDirectory 9.3 on Windows operating system can be made silent and more flexible by using a response file for the following:

- Complete unattended installation with all required user inputs
- Default configuration of components
- Bypassing all prompts during the installation

A response file is a text file containing sections and keys, similar to a `Windows.ini` file. You can create and edit a response file using any ASCII text editor. The eDirectory upgrade reads the installation parameters directly from the response file and replaces the default installation values with response file values. The installation program accepts the values from the response file and continues to install without prompts.

Response File Sections and Keys

The eDirectory 9.3 installation requires changes to the sections in the response file to add information about the OpenText eDirectory instance to be installed, including the tree name, administrator context, administrator credentials (including user name and passwords), installation locations, etc. A full list of the keys and their default values is available in the sample response files which are delivered with the eDirectory installation. There are four response files available at `<eDirectoryInstallPath>\NetIQ\eDirectory\Sample_Response_File` during the OpenText eDirectory installation:

- `newtree.ni`: This file is used to configure a new OpenText eDirectory tree.
- `existingtree.ni`: This file is used to add a server to an existing OpenText eDirectory tree.
- `upgrade.ni`: This file is used to upgrade the OpenText eDirectory server.
- `deconfigure.ni`: This file is used to de-configure an OpenText eDirectory tree.

NOTE: You should use any of the provided response files during the OpenText eDirectory installation.There are essential parameters and set by default in these files. When editing these files, ensure that there are no blank spaces between the key and the values along with the equals sign ("=") in each key-value pair.

Adding Features to the Automated Installation

Most details for configuring the eDirectory Installer have default setting for the manual installation. However, during unattended installation, each configuration parameter must be explicitly configured. This section discusses the basic settings to be configured, irrespective of any sequence of installation or additional features.

OpenText eDirectory Server Details

Regardless of whether it is an upgrade or a primary/secondary server installation, the details of the server being installed or upgraded must be provided to the Installer. Most of this information is configured in the tag `[NWI:NDS]`.

`[NWI:NDS]`

- **mode:** By default, the mode key is set to configure. This configures OpenText eDirectory.
- **Tree Name:** For a primary server installation, this is the name of the tree that needs to installed. For a secondary server installation, this is the tree to which this server must be added.
- **Server Name:** The name of the server that is being installed.
- **Server Container:** Any server added to a tree has a server object containing all the configuration details specific to the server. This parameter is the container object in the tree to which the server object will be added. For primary server installations, this container will be created with the server object.
- **Admin Login Name:** The name (RDN) of the Administrator object in the tree that has full rights, at least to the context to which this server is added. All operations in the tree will be performed as this user.
- **Admin Context:** Any user added to a tree has a user object that contains all the user-specific details. This parameter is the container object in the tree to which the Administrator object will be added. For primary server installations, this container will be created with the server object.

- **Admin password:** The password for the Administrator object created in the previous parameters. This password will be configured to the Administrator object during primary server installations. For secondary server installations, this needs to be the password of the Administrator object in the primary server that has rights to the context to which the new server is added.

  We recommend you to set the admin password in an environment variable and mention the environment variable name in the response file. Once the silent configuration is complete, remove the password from the environment variable.

  IMPORTANT: You provide the administrator user credentials in the response file for an unattended installation. Therefore, you should permanently delete the file after the installation to prevent the administrator credentials from being compromised.

- **DataDir:** By default the DIB is installed in the `Files` subfolder inside the NDS location, but administrators can change this parameter and provide a different location. If no value is provided for this parameter, the value will be set to `<Install location>/DIBFiles` by default.
- **EBA:** Enhanced Background Authentication (EBA) provides an improved and more secure background authentication protocol for authenticating to the NCP servers in the tree. OpenText eDirectory provides the option of enabling EBA while configuring the OpenText eDirectory tree or later. By default, EBA is not configured on OpenText eDirectory unless it is changed in the response file. To enable EBA, set Require EBA to Yes.
- **FIPS:** OpenText supports eDirectory running in Federal Information Processing Standard (FIPS) mode. To enable OpenText eDirectory in FIPS mode, set Require FIPS for TLS to Yes.
- **Enable PBKDF2:** A new configuration parameter `Enable` PBKDF2 has been added to the `newtree.ni` response file in OpenText eDirectory 9.3. If this option is set to yes, a password policy is created and assigned automatically to the whole tree. This password policy enables synchronization of NDS passwords with PBKDF2 passwords for all users in the tree. For more information, see Understanding Non-Reversible Password Storage in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.

The following is a sample of text in the response file for all the basic parameters described above:

```
[NWI:NDS]
mode=configure
New Tree=Yes
Tree Name=ENEWTREE
Server Name=ENEWSERVER
Server Container=myorg
Admin Context=myorg
Admin Login Name=Admin
Admin Password=env: PASSWORD_VAR
Require IPV6=NO
Require EBA=NO
Require FIPS for TLS=NO
DataDir=C:\NetIQ\eDirectory\DIBFiles
LDAP TCP Port=389
LDAP SSL Port=636
Require TLS=No
Require SS=YES
Enable PBKDF2=No
```

Adding NMAS Methods

OpenText eDirectory supports installation of multiple NMAS methods, both during install and upgrade. During manual installations, you can select the NMAS methods to install and configure. This can also be achieved in automated installations.

The NMAS-related configuration settings are provided inside the `[NWI:NMAS]` tag. The tag has two keys to be configured, and both are mandatory:

- **Choices:** This key informs the OpenText eDirectory installation component on the number of NMAS methods that need to be installed.
- **Methods:** This key lists the NMAS method options that need to be installed. Currently, there are 6 supported NMAS methods. The method names and their types are as follows:

Table 4-1 *NMAS Methods*

Method Name	Method Type
CertMutual	Certificate mutual login method
Challenge Response	The OpenText challenge response NMAS method
DIGEST-MD5	Digest MD5 login method
SAML	Security Assertion Markup Language authentication method
NDS	NDS login method (default)
Simple Password	Simple password NMAS login method
SCRAM	The Salt Challenge Response Authentication Mechanism (SCRAM) uses the PBKDF2 hash based passwords.

NOTE: The method names should exactly match those listed in the above table, as options to the Methods key. The Installer matches the exact string (with case) for choosing the NMAS methods to install.

The NDS NMAS method is mandatory and will be installed automatically if no NMAS methods list is provided. However, if you are creating an explicit list, do not remove this method from the list.

If the NMAS methods are configured using this methodology in the response file, OpenText eDirectory shows a status message while installing, without prompting for user input.

The following is sample text in the response file for choosing the NMAS methods:

```
[NWI:NMAS]
Methods=CertMutual,Challenge Response,DIGEST-MD5,NDS,Simple Password,SAML
```

HTTP Ports

OpenText eDirectory listens on preconfigured HTTP ports for access through the Web. For example, OpenText iMonitor accesses eDirectory through Web interfaces. They need to specify certain in order to access the appropriate applications. There are two keys that can be set prior to installation to configure eDirectory on specific ports:

- **Clear Text HTTP Port:** The port number for the HTTP operations in clear text.
- **SSL HTTP Port:** HTTP port number for operations on the secure socket layer.

The following is sample text in the response file for configuring HTTP port numbers:

```
[eDir:HTTP]

Clear Text HTTP Port=8028

SSL HTTP Port=8030
```

LDAP Configuration

OpenText eDirectory supports LDAP operations. It listens for LDAP requests in clear text and SSL, on two different ports. These ports can be configured in the response file prior to installation so that when OpenText eDirectory is started, it listens on these configured ports.

There are three keys in the `[NWI:NDS]` tag that configure the LDAP ports:

- **LDAP TCP Port:** The port on which OpenText eDirectory should listen for LDAP requests in clear text. If no port is mentioned, 389 will be assumed by default.
- **LDAP SSL Port:** The port on which OpenText eDirectory should listen for LDAP requests in SSL. You can also use a key to configure whether OpenText eDirectory should mandate secure connections when bind requests send the password in clear text. If no port is mentioned, 636 will be assumed by default.
- **Require TLS:** Whether OpenText eDirectory should mandate TLS when receiving LDAP requests in clear text. If no value is provided for this parameter, by default it will be set to `Yes`.

The following is sample text in the response file for LDAP configuration:

```
[NWI:NDS]

Require TLS=Yes

LDAP TLS Port=389

LDAP SSL Port=636
```

Controlling Automated Installation

The response file can also be edited to control the flow of automated installation.

Stopping SNMP services

This feature is specific to an OpenText eDirectory installation on Windows. Most Windows servers have SNMP configured and running. When OpenText eDirectory installs, the SNMP services need to be brought down and restarted after the installation. With manual installations, the Installer prompts the user on-screen to stop the SNMP services before continuing the installation. This prompt can be avoided during automation by setting the key in the `[NWI:SNMP]` tag:

- **Stop service:** Set the value to Yes to stop the SNMP services without prompting. The status of is displayed on-screen.

The following is sample text in the response file for stopping SNMP services:

```
[NWI:SNMP]

Stop service=yes
```

Specifying Default Parameters for Default Server Certificates

OpenText eDirectory provides the option to specify the default RSA key size, Elliptic Curve and certificate life for the CA certificates and default server certificates while configuring a new OpenText eDirectory tree. You can specify the following default parameters for the CA and default server certificates during silent installation of a new eDirectory tree in the response file:

- **RSA Key Size:** To specify the key size for RSA certificates. Allowed values are 2048, 4096 and 8192 bits.
- **EC Curve:** To specify the curve limit for EC certificates. Allowed values are P256, P384 and P521.
- **Certificate Life:** To specify the certificate life in number of years.

The values specified here will be set on corresponding attributes on the Organizational CA object when the new tree is configured.

These attributes can be set in the [NWI:PKI] tag of the `newtree.in` file while installing a new eDirectory server, as shown in the below sample:

```
[NWI:PKI]
RSA KeySize=4096
EC Curve=P521
Certificate Life=4
```

For more information, see Creating an Organizational Certificate Authority Object in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Primary/Secondary Server Installation

OpenText eDirectory Installer provides options for the unattended install of a primary or a secondary server, into a network. There is one key that help the Installer decide whether it is a primary or a secondary server installation.

- **Primary Server:** Use `New Tree` key in the `[NWI:NDS]` tag and set it to `Yes` for a new/primary tree installation in the `newtree.ni` file or in a similar response file which is required for setting up a new server.
- **Secondary Server:** Use `New Tree` key in the `[NWI:NDS]` tag and set it to `No` for a secondary tree installation in the `existingtree.ni` file or in a similar response file which is required for setting up a secondary server.

For example, the keys for installing a primary server in a new tree would be as follows:

```
[NWI:NDS]
New Tree=Yes
```

and for a secondary server installation into an existing tree:

```
[NWI:NDS]
New Tree=No
```

Unattended Installation of OpenText eDirectory using Response File

Launching the OpenText eDirectory Installer on Windows is easy. The `eDirectory_930_Windows_x86_64.exe` delivered in the eDirectory release is invoked in the command line with a few additional parameters.

Depending on the setup mode you have mentioned, use either of the following commands:

Install

Run the following command in Windows command prompt:

`<Download Location Path>\eDirectory_930_Windows_x86_64.exe /qn`

For example, `D:\builds\eDirectory_930_Windows_x86_64.exe /qn`

NOTE: Run the following command to install OpenText eDirectory in custom location:

`eDirectory_930_Windows_x86_64.exe /qn INSTALLDIR="C:\<Install Location>`

Configure

Run the following command in Windows PowerShell:

```
<eDirectory installed location> ./EConfig.ps1 -rfile
<Sample_Response_Files location>\newtree.ni
```

For example,

```
C:\NetIQ\eDirectory> ./EConfig.ps1 -rfile
C:\Sample_Response_Files\newtree.ni
```

NOTE: The log files can be accessed from the following locations:

- `C:\Program Files\NetIQ\eDirectory\installlogs`
- `C:\Program Files\NetIQ\eDirectory\logs`

Locating Log Files

dsinstall.log

The first part of the `dsinstall.log` file available at `<Windows Drive>\NetIQ\eDirectory` lists environment variables that are set. The second part contains status messages documenting the OpenText eDirectory installation process.

Upgrading OpenText eDirectory on Windows

It is recommended not to directly upgrade eDirectory from version 8.8SP1 to 9.3.You must upgrade to 9.2 and then to 9.3. If you have an EBA enabled server then you must be on OpenText eDirectory 9.2.8 or 9.2.9 before upgrading to OpenText eDirectory 9.3.0.

If you have NMAS method installed already, then to upgrade OpenText eDirectory 9.30 it is required to extend the NMAS schema before upgrade. For more information see, "NMAS Schema Extension before Upgrading OpenText eDirectory" on page 83.

NOTE: To upgrade from a 32-bit version of OpenText eDirectory to a 64-bit version of OpenText eDirectory, first upgrade 32-bit version to eDirectory 8.8.x 64-bit version and then upgrade it to OpenText eDirectory 9.2. You can follow the same procedure for upgrading a 64-bit OpenText eDirectory to eDirectory 9.3.

The following sections provide information to help you upgrade your existing OpenText eDirectory installation to the current version.

- "NMAS Schema Extension before Upgrading OpenText eDirectory" on page 83
- "Upgrading OpenText eDirectory Using Windows Installer" on page 84
- "Unattended Upgrade of OpenText eDirectory on Windows" on page 85

NMAS Schema Extension before Upgrading OpenText eDirectory

When `eDirectory_930_Windows_x86_64.exe` is downloaded and extracted, there is a `preUpgrade_930` file present along with the installer. Navigate to the `preUpgrade_930` to see the below files.

- `preUpgrade_930/preUpgrade.ps`
- `preUpgrade_930/upgrade.ni`
- `preUpgrade_930/nmas.sch`

Schema extension in windows:

Before upgrading OpenText eDirectory to 9.3.0, follow the below steps to extend the schema.

1. Update the `upgrade.ni` file present in preUpgrade_930 folder. Mention the tree name, server name and admin credential of the existing eDirectory server in the `upgrade.ni` response file. Below is a sample `upgrade.ni` response file shown with the upgrade configuration.

```
[NWI:NDS]
mode=configure
Tree Name=enewtree
Server Name=enewserver
Server Container=org
Admin Context=org
Admin Login Name=Admin
Admin Password=env:PASSWORD_VAR
Require IPV6=NO
Require EBA=NO
Require FIPS for TLS=YES
LDAP TCP Port=389
LDAP SSL Port=636
Require TLS=Yes
Require SS=Yes
Existing Server=10.10.10.10
Existing Server Port=524
[NWI:SNMP]
Stop service=No
[NWI:NMAS]
Methods=CertMutual,Challenge Response,DIGEST-MD5,NDS,Simple
Password,SAML
```

2 Run the below command from powershell.

```
preUpgrade.ps1 -rfile <path_for_Configuration>/upgrade.ni
```

3 Once the schema extension is successful, continue to upgrade eDirectory to 9.3.0.

Upgrading OpenText eDirectory Using Windows Installer

You can upgrade your OpenText eDirectory server using the Windows installer. Perform the following steps to upgrade your eDirectory server:

1 At the Windows server, log in as Administrator or as a user with administrative privileges

2 Run `eDirectory_930_Windows_x86_64.exe` from the windows folder in the eDirectory 9.3 CD or from the downloaded file.

3 The installer screen will now display the existing OpenText eDirectory tree name and server FDN on the **Basic** tab. Enter the tree admin credentials and click on the **Upgrade** button to proceed with the upgrade process.

4 On the **Advanced** tab, you can change the existing settings that was set while installing OpenText eDirectory. For more information, see "Installing or Updating OpenText eDirectory 9.3 on a Windows Server" on page 72.

NOTE: To upgrade the Replica Server with EBA in Windows for eDirectory 9.2.8 and 9.2.9, see "Installing Bridge Patch to Multiple Replica Servers with EBA in Windows" on page 25

Unattended Upgrade of OpenText eDirectory on Windows

Upgrading OpenText eDirectory on Windows can be done in silent mode.

On Windows, you must install the latest version of OpenText eDirectory before upgrading using following command:

```
<Download Location Path>\eDirectory_930_Windows_x86_64.exe /qn
```

For example,

```
D:\builds\eDirectory_930_Windows_x86_64.exe /qn
```

For more information, see "Unattended Install and Configure to OpenText eDirectory 9.3 on Windows" on page 76.

After upgrading all the libraries successfully, mention the tree name, server name and admin credential of the existing eDirectory server in the `upgrade.ni` response file.

Below is a sample `upgrade.ni` response file shown with the upgrade configuration:

```
[NWI:NDS]
mode=configure
Tree Name=enewtree
Server Name=enewserver
Server Container=org
Admin Context=org
Admin Login Name=Admin
Admin Password=env:PASSWORD_VAR
Require IPV6=NO
Require EBA=NO
Require FIPS for TLS=YES
LDAP TCP Port=389
LDAP SSL Port=636
Require TLS=Yes
Require SS=Yes
Existing Server=10.10.10.10
Existing Server Port=524

[NWI:SNMP]
Stop service=No

[NWI:NMAS]
Methods=CertMutual,Challenge Response,DIGEST-MD5,NDS,Simple Password,SAML
```

Once the `upgrade.ni` response file is updated with the required OpenText eDirectory server details, run the following command to upgrade your OpenText eDirectory server:

```
<eDirectory installed location> ./EConfig.ps1 -rfile
<Sample_Response_Files location>\upgrade.ni
```

For example, `C:\NetIQ\eDirectory> ./EConfig.ps1 -rfile C:\Sample_Response_Files\upgrade.ni`.

5 Deploying OpenText eDirectory on Microsoft Azure

OpenText eDirectory can be deployed on Microsoft Azure virtual machines. For more information on the supported operating systems, see “System Requirements” on page 27 for Linux or “System Requirements” on page 67 for Windows.

Prerequisites

In addition to the system requirements of OpenText eDirectory, ensure that you meet the following requirements:

- An administrative account on Azure.
- The OpenText eDirectory installer (tarball) has been downloaded, extracted, and available for copying to the virtual machines. For Windows, download the Windows executable file.
- An SSH client to connect to the Azure virtual machines from the client machine.

Deployment Procedure

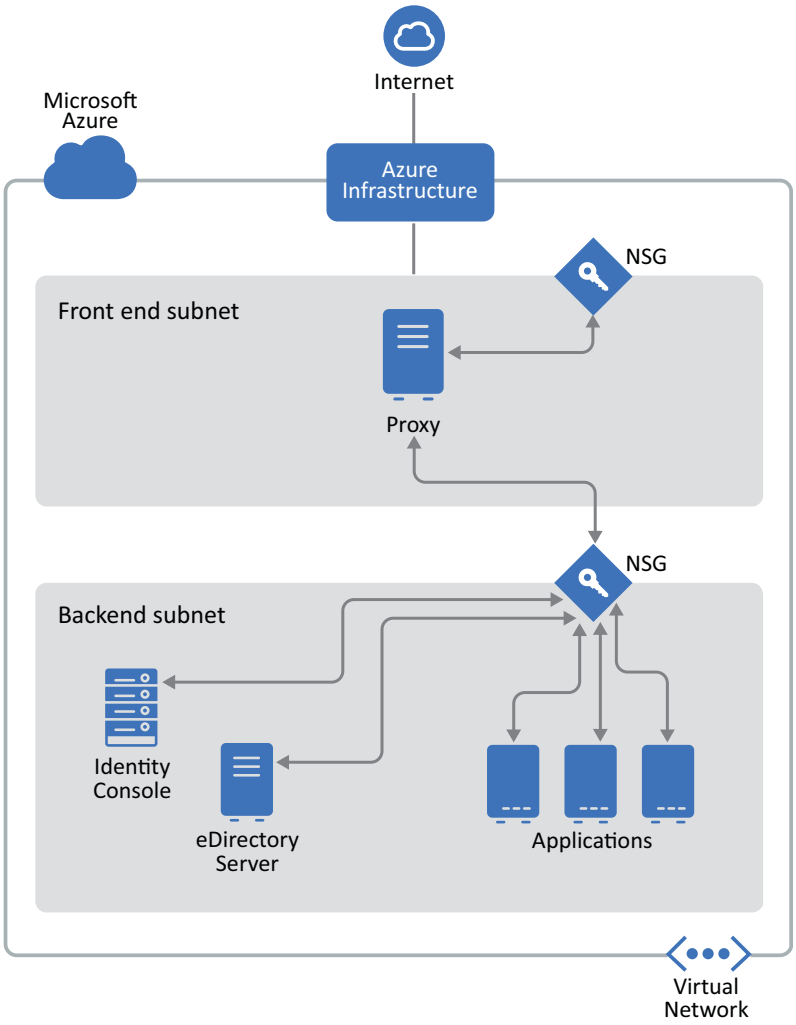
The following section provides instructions for deploying OpenText eDirectory on Microsoft Azure depending on the operating system running on the virtual machines.

- “Deploying on a Linux Platform” on page 87
- “Deploying on a Windows Platform” on page 101

Deploying on a Linux Platform

OpenText eDirectory should be deployed only in a backend subnet in Azure Virtual Network. Figure 5-1 illustrates a sample deployment that is used in the subsequent sections.

Figure 5-1 *OpenText eDirectory Deployment on Azure*



NOTE: ◆Proxy is bastion host in the front end subnet to which the administrator connects using SSH and connects to other instances in the backend subnet using the SSH agent forwarding.

- Applications that need to access OpenText eDirectory should be deployed in the backend subnet. If these applications need to be accessed from the Internet, configure an Azure load balancer in the front end subnet to enable the access. For more information, see Create a Public Basic Load Balancer.
- The Hybrid Set-up i.e installing the OpenText eDirectory Master on-premises and replica on Azure Virtual Machine and vice versa is not supported.

The deployment procedure consists of the following steps:

- "Preparing Azure Services" on page 89
- "Configuring Application Security Groups (ASG)" on page 89
- "Configuring Network Security Groups (NSG) for Subnets" on page 90
- "Configuring Network Security Groups for Virtual Machine" on page 92
- "Creating a SSH Key Pair" on page 94

- “Creating and Deploying Virtual Machines” on page 94
- “Configuring Data Disk for Storing OpenText eDirectory Data” on page 95
- “Installing OpenText eDirectory and OpenText Identity Console” on page 96
- “Deploying Auditing Services” on page 99
- “Disaster Recovery” on page 100

Preparing Azure Services

This section outlines general steps for creating Azure services for use with OpenText eDirectory. This includes creating the resource groups, virtual network (VNet) and subnets.

IMPORTANT: While creating services (such as, virtual network, security groups, virtual machines, etc.), ensure to specify the same value for **Location**.

Creating Resource Groups

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. For example, while deploying OpenText eDirectory on Azure, the resource groups should contain Virtual Machines, Virtual Network, Application Security Groups, Network Security Groups, Public IP Address, Network Interface and Disks. For more information on how to create a resource group, see Manage Azure resources through portal.

NOTE: Not all administrators may have rights to create a new resource group.

Creating a Virtual Network

Azure Virtual Network enables many types of Azure resources, such as Azure Virtual Machines (VM), to securely communicate with each other within the Azure Internet, and on-premises networks. For more information, see What is Azure Virtual Network.

As part of creating the virtual network, one subnet gets created by default. If you want to create multiple subnets, go to the newly created **Virtual Network** > **Subnet** > **Add Subnet**.

Configuring Application Security Groups (ASG)

Application security groups enable you to configure network security as a natural extension of an application's structure. It also allows you to group virtual machines and define network security policies based on those groups. For more information, see Application Security Groups.

You must create the following Application security groups before configuring the network security groups:

Table 5-1 *Application Security Groups*

Name	Description
SSH_Proxy	Contains the network interface of the virtual machine where SSH_Proxy will be configured
eDirectory	Contains the network interface of all the virtual machines where OpenText eDirectory will be configured
eDirectory_CA	Contains the network interface of the virtual machine where OpenText eDirectory server hosting the tree CA will be configured
OpenText Identity Console	Contains the network interface of the virtual machine where OpenText Identity Console will be configured

Configuring Network Security Groups (NSG) for Subnets

You can filter inbound and outbound network traffic for subnet with an NSG. NSGs contain security rules that filter network traffic by IP address, port, and protocol.

This section outlines the rules to create an NSG for the front end subnet. Configure the following rules over the default security rules:

- Inbound Rules:

Table 5-2 *Inbound Rules for the Front End Subnet*

Priority	Name	Port Range	Source	Destination	Action	Description
100	SSH	TCP 22	Any	SSH_Proxy (ASG)	ALLOW	Allows SSH connection to the Proxy server from the Internet
110	Allow Subnet Traffic	Any	Any	Front end subnet	ALLOW	(Optional) Allows all intra-subnet traffic. **NOTE:** Set this rule only if you have other virtual machines in your front end subnet which need to communicate with each other.

Priority	Name	Port Range	Source	Destination	Action	Description
120	All Traffic	All	Any	Any	DENY	Denies all inbound traffic not already handled by a preceding rule

This section outlines the rules to create NSGs in the backend subnet. Configure the following rules for network security groups:

- Inbound Rules:

Table 5-3 *Inbound Rules for the Backend Subnet*

Priority	Name	Port Range	Source	Destination	Action	Description
100	SSH	TCP 22	Proxy (ASG)	Backend subnet	ALLOW	Allows inbound SSH traffic from the SSH Proxy
110	Identity Console	TCP 9000	Proxy (ASG)	OpenText Identity Console (ASG)	ALLOW	Allows HTTPS traffic for accessing OpenText Identity Console from SSH Proxy
120	HTTP CRL	TCP 8028	Virtual Network	eDirectory_CA (ASG)	ALLOW	Required for accessing the OpenText eDirectory tree CRL from the VNet when there are services in the VNet which are configured with certificates issued by tree CA.
130	Allow Subnet Traffic	Any	Backend subnet	Backend subnet	ALLOW	Allows all intra-subnet traffic
140	All Traffic	All	Any	Any	DENY	Denies all inbound traffic

Configuring Network Security Groups for Virtual Machine

A security group is a set of virtual firewall rules which can be assigned to one or more virtual machines in the Virtual Network.

By default, a new security group only allows incoming traffic on port 22, so that you can only connect to the instance by using SSH.

For more information, see Security Groups.

To deploy OpenText eDirectory on Azure, create the following network security groups: eDirectory_NSG_1, eDirectory_NSG_2 and IdentityConsole_NSG. Create these security groups with the following port rules over the default security rules:

1. **eDirectory_NSG_1:** This NSG should be associated with the virtual machine hosting the eDirectory tree CA.

Priority	Name	Port Range	Source	Destination	Action	Description
100	SSH	TCP 22	SSH Proxy (ASG)	OpenText eDirectory (ASG)	ALLOW	Allows SSH traffic from SSH Proxy
110	NCP	TCP 524	Backend subnet	OpenText eDirectory (ASG)	ALLOW	Allows NCP traffic for OpenText eDirectory in backend subnet
120	HTTP CRL	TCP 8028	Virtual Network	eDirectory_CA (ASG)	ALLOW	Required for accessing the eDirectory OpenText tree CRL from the VNet when there are services in the VNet which are configured with certificates issued by tree CA.
130	LDAPS	TCP 636	Backend subnet	OpenText eDirectory (ASG)	ALLOW	Allows secured LDAP traffic in backend subnet

Priority	Name	Port Range	Source	Destination	Action	Description
140	SLP	Any 427	Backend subnet	OpenText eDirectory (ASG)	ALLOW	Allows SLP traffic in the backend subnet
150	All Traffic	All	Any	Any	DENY	Denies all inbound traffic

NOTE: OpenText eDirectory servers should not be configured to listen on LDAP port 389 and access to port 389 should not be allowed on the security group which is assigned to OpenText eDirectory. Also access to HTTP port should only be allowed on the security group assigned to OpenText eDirectory server hosting the tree CA.

2. **eDirectory_NSG_2:** This NSG should be associated with all virtual machines hosting OpenText eDirectory servers other than the OpenText eDirectory tree CA.

Priority	Name	Port Range	Source	Destination	Action	Description
100	SSH	TCP 22	SSH Proxy (ASG)	OpenText eDirectory (ASG)	ALLOW	Allows SSH traffic from SSH Proxy
110	NCP	TCP 524	Backend subnet	OpenText eDirectory (ASG)	ALLOW	Allows NCP traffic for OpenText eDirectory in backend subnet
120	LDAPS	TCP 636	Backend subnet	OpenText eDirectory (ASG)	ALLOW	Allows secured LDAP traffic in backend subnet
130	SLP	Any 427	Backend subnet	OpenText eDirectory (ASG)	ALLOW	Allows SLP traffic in the backend subnet
140	All Traffic	All	Any	Any	DENY	Denies all inbound traffic

3. **IdentityConsole_NSG:** This NSG should be associated with the virtual machine hosting Identity Console. The following NSG rules enable access to the Identity Console server from the Proxy server only.

Priority	Name	Port Range	Source	Destination	Action	Description
100	SSH	TCP 22	SSH Proxy (ASG)	OpenText Identity Console (ASG)	ALLOW	Allows SSH traffic from Proxy
110	HTTPS	TCP 9000	SSH Proxy (ASG)	OpenText Identity Console (ASG)	ALLOW	Allows secured HTTP traffic for accessing Identity Console from Proxy
120	All Traffic	All	Any	Any	DENY	Denies all inbound traffic

Creating a SSH Key Pair

You must create a SSH key pair before configuring the Azure VMs. To create a key pair, perform the following steps:

1 Create a 4096-bit RSA SSH key pair on your client using the following command:

```
ssh-keygen -t rsa -b 4096
```

`ssh-keygen` places the newly created public key at `~/.ssh/id_rsa.pub`.

2 Provide the above SSH public key to your Azure account. For more information, see Provide an SSH Public Key.

IMPORTANT: You can connect to and manage your virtual machines only using the SSH private key. Therefore, do not lose the SSH private key.

Creating and Deploying Virtual Machines

Create and launch your virtual machines (VM) on one of the supported platforms. For more information on how to create and launch VMs, see Create and Launch Your Linux Virtual Machine. As a part of creating and launching the instances, you must also perform the following steps:

1 Associate eDirectory_NSG_1 with the VMs where the first OpenText eDirectory server will be configured, eDirectory_NSG_2 with the VM where all other OpenText eDirectory servers will be configured and IdentityConsole_NSG with the VM where Identity Console will be configured. For more information about security groups, see “Configuring Network Security Groups for Virtual Machine” on page 92.

2 Associate the public key created in section “Creating a SSH Key Pair” on page 94 with your instances.

NOTE: When multiple Availability Zones are available for the selected Azure Location, the replica servers should not be deployed in the same Availability Zone as the master OpenText eDirectory server.

Configuring Data Disk for Storing OpenText eDirectory Data

Configuring data disk is required to prevent loss of OpenText eDirectory data and configuration in case of Azure vm crash. For more information on recovering OpenText eDirectory data and configuration, see "Disaster Recovery" on page 100. After creating the VM, perform the following steps to prepare the VM for deploying OpenText eDirectory:

1. Create and attach a data disk, perform the steps in Use the portal to attach a data disk to a Linux VM.
2. Login to the VM, format the data disk with `ext4` file system and mount the data disk. For more information on how to format and mount the data disk, see Connect to the Linux VM to mount the new disk.
3. Bind mount directories from the data disk to OpenText eDirectory data/NICI data directories. Perform the following steps as root user to bind mount:

   3a Create OpenText eDirectory data directory by using the following command:

   ```
   mkdir <mount_point>/eDirectory_data
   ```

   3b Create NICI data directory by using the following command:

   ```
   mkdir <mount_point>/nici_data
   ```

   3c Create NICI and OpenText eDirectory configuration directories by using the following command:

   ```
   mkdir <mount_point>/eDirectory_nici_conf
   ```

   3d Create required directories for OpenText eDirectory by using the following commands:

   ```
   mkdir --parents /var/opt/novell/eDirectory
   mkdir –-parents /var/opt/novell/nici
   mkdir –-parents /etc/opt/novell/eDirectory
   ```

   3e To bind mount the directories, add the following to `/etc/fstab`:

   ```
   <mount_point>/eDirectory_data /var/opt/novell/eDirectory none
   defaults,bind 0 0

   <mount_point>/nici_data /var/opt/novell/nici none defaults,bind 0 0

   <mount_point>/eDirectory_nici_conf /etc/opt/novell/eDirectory none
   defaults,bind 0 0
   ```

NOTE: All operations in the VM should be performed as a root user.

Installing OpenText eDirectory and OpenText Identity Console

Prerequisites

- ❒ Ensure that you meet the requirements listed in "System Requirements" on page 27.
- ❒ Create security groups as mentioned in "Configuring Network Security Groups for Virtual Machine" on page 92.
- ❒ Proxy VM should be hardened and secured server. SSH private key required for accessing VMs in the backend subnet and the Proxy VM, should not be stored in the VNet but on the client only. Choose a VM size that provides good performance and memory for this instance.
- ❒ Create an additional network interface for the Proxy VM and assign a static public IP address to that interface.
- ❒ Configure VNC server in the Proxy VM. VNC server should be hardened with a password of good strength. Connect to the VNC server through an SSH tunnel to allow secured communication. VNC server should be configured to listen only for connections from localhost. Disable screen lock to avoid session lockout. After using the VNC server, you should terminate the session.
- ❒ Update the `/etc/hosts` file of VMs manually with `IP-Address Full-Qualified-Hostname Short-Hostname` entry. This is to work around the limitation with Azure to perform a reverse DNS lookup.
- ❒ Connect to the VM in the backend subnet where OpenText eDirectory/OpenText Identity Console will be configured using SSH proxy:

  ```
  ssh -i edir_key.pem -A -J azureuser@<ssh_proxy_ip>
  azureuser@<instance_private_ip>
  ```

  NOTE: ◆in above sample commands, `edir_key.pem` is a sample file name containing the server key.

  - ◆ You can also add the identity file in the agent using the SSH-Add command to avoid using identity file every time you login.

  To view the private IP address of a VM, click **Instances** > *[instance]* > **Description**.
- ❒ Configure an SLP Directory Agent (DA) server in a VM in the Backend subnet. Open port 427 in the inbound rule of NSG for the VM where SLP DA is deployed. Enable DA operation by editing the `slp.conf` file. For more information, see Configuring OpenSLP for OpenText eDirectory in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Installation and Configuration Procedure

This section explains the step by step instructions to install and configure OpenText eDirectory and OpenText Identity Console in an Azure environment. Once OpenText eDirectory is installed, you should ensure that the following conditions are met:

- ◆ EBA is enabled
- ◆ SNMP is disabled
- ◆ OpenText eDirectory is not listening on port 389
- ◆ LDAP and HTTP services are configured to use ECDSA certificates only

- Access to the SSH port of the Azure VMs in the backend subnet should be disabled when not in use
- Disable OpenText iMonitor, eMBox and DHost modules to provide additional security. After disabling them, all activities involving these modules should be performed using NDS utilities only.

Installing & Configuring OpenText eDirectory

1. Copy the `eDirectory_<version>_Linux_x86_64.tar.gz` file using Secure Copy (scp) to the VM in the backend subnet where OpenText eDirectory will be configured using SSH proxy:

   ```
   scp -i <keyname> -o ProxyJump=vm-user@<ssh_proxy_ip>
   eDirectory_<version>_Linux_x86_64.tar.gz vm-user@<instance_ip>:/
   <directory>
   ```

2. Install OpenText eDirectory. For more information, see Using the nds-install Utility to Install OpenText eDirectory Components.
3. Configure OpenText eDirectory. For more information, see Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server. For example, here's a sample command for installing and configuring OpenText eDirectory:

   ```
   ndsconfig  new [-t <tree_name>] [-n <server context>] -a <admin FDN> [-
   w <admin password>] -P ldaps://<instance_ip> --configure-eba-now yes
   ```

4. Install `openslp-server` and start the SLPD service.

Installing & Configuring Identity Console

Using the Identity console, you can manage the OpenText eDirectory operations on your Azure environment. OpenText Identity Console should be installed on your Azure VM after installing OpenText eDirectory.

1. Copy the `IdentityConsole_170_Containers.tar.gz` file using Secure Copy (scp) to the instance in the backend subnet where OpenText Identity Console will be configured using SSH proxy:

   ```
   scp -i <keyname> -o ProxyJump=vm-user@<ssh_proxy_ip>
   IdentityConsole_170_Containers.tar.gz vm-user@<instance_ip>:/<directory>
   ```

2. Install and configure OpenText Identity Console. For more information, see Installing OpenText Identity Console Workstation on Windows . Before installing OpenText Identity Console, see the system requirements in System Requirements and Prerequisites for Workstation section in the OpenText™ Identity Console Installation Guide.
3. Replace the Self-Signed certificates in the VM running Identity Console with a secure CA signed certificates. For more information, see Replacing the Temporary Self-Signed Certificates for OpenText iManager.

Launching OpenText Identity Console

Perform the following steps to launch OpenText Identity Console:

1. Connect to the VNC server running on localhost of Proxy VM through SSH tunnel.
2. Install and launch a browser in the same instance.
3. Launch and connect to the OpenText eDirectory tree using the IP address or the tree name.

Post-Configuration Tasks

1. To check if EBA is enabled, see Viewing Information About EBA in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
2. Enable Suite B on Certificate Server. For more information, see Enabling Suite B on the Certificate Server in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
3. Configure AES 256-bit tree key for the first OpenText eDirectory server. For more information, see Creating an AES 256-Bit Tree Key in the *NICI Administration Guide*.
4. Delete the CRL distribution points in the first OpenText eDirectory server. As non-secured LDAP access over port 389 is disabled on all OpenText eDirectory servers, the CRL for the tree CA should be available for download over HTTP only. Perform the following steps to delete the CRL distribution points:
   - **4a** Login to OpenText Identity Console as an Administrator.
   - **4b** Go to **Certificate Management** > **CA Management** > **CRL** > Create CRL .
   - **4c** Click **One**. Select and delete all **CRL Distribution Points** except the HTTP CRL Distribution Point (http://<instance_ip>:8028/crl/one.crl).
   - **4d** Click **Apply** and then click **Close**.
   - **4e** Click **OneEC**. Select and delete all **CRL Distribution Points** except the HTTP CRL Distribution Point (http://<instance_ip>:8028/crl/oneec.crl).
   - **4f** Click **Apply** and then click **OK**.
5. Repair the server's default certificates using the OpenText Identity Console certificate management. To repair the default certificates, perform the following steps:
   - **5a** Login to OpenText Identity Console as Administrator.
   - **5b** Go to **Certificates Management** > **Default Certificate**.
   - **5c** Select the server(s) which owns the certificates and click **Next**.
   - **5d** Select **Yes All Default Certificates will be overwritten** and click **Next**.
   - **5e** Review the tasks to be performed and select **Finish** > click **OK**.
6. Configure LDAP and HTTP services to use ECDSA Certificates and Suite B ciphers. For more information, see Configuring LDAP and HTTP Services to Use ECDSA Certificates and Suite B Ciphers in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*. Once done, restart OpenText eDirectory.
7. For more information to check if SNMP sub-agent is unloaded, see Loading and Unloading the SNMP Server Module in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
8. Ensure that OpenText eDirectory is not listening on port 389.
9. Disable OpenText iMonitor, eMBox, DHost and HTTP stack.
   - **9a** Perform the following steps to disable iMonitor, eMBox and DHost in the OpenText eDirectory server hosting the tree CA:
     - **9a1** Edit the `ndsmodules.conf` file by commenting `hconserv`, `imon` and `embox`.
     - **9a2** Restart OpenText eDirectory.

9b Perform the following steps to disable the HTTP stack in the OpenText eDirectory replica servers:

9b1 Edit the `ndsmodules.conf` file by commenting `httpstk`, `hconserv`, `imon` and `embox`.

9b2 Restart eDirectory.

NOTE: `httpstk` should be placed above `nds` in the `ndsmodules.conf` file before commenting. This stops `nds` module from enabling HTTP stack.

10 Configure SLP to force OpenText eDirectory to use unicast as advertising method. Edit the `slp.conf` file by providing the IP address of the DA server in the Backend subnet. For more information, see Configuration Parameters in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

NOTE: Once all OpenText eDirectory VMs and OpenText Identity Console have been configured, configure the security rules of the Azure backend subnet to deny access to the SSH port and allow it only when required.

Deploying Auditing Services

You can deploy the Common Event Format (CEF) auditing service on Azure to audit various OpenText eDirectory events. Perform the following steps to deploy CEF auditing services:

1. Install an auditing server in the VNet.
2. Configure the auditing server to listen on a port

   NOTE: We recommend you to use Sentinel as your auditing server.

3. Create a new network security group rule in the front end subnet with the following configuration and associate with the VM where the audit server is running:

Name	Port	Source	Destination	Description
Auditing Server Port	TCP (Auditing server port)	Backend subnet	Auditing server IP	Allows receiving events from OpenText eDirectory servers

4. Update the following in `/etc/opt/novell/eDirectory/conf/auditlogconfig.properties` file on all the OpenText eDirectory instances:

   ```
   log4j.appender.S.Host=<Auditing server ip>
   log4j.appender.S.Port=<auditing server port>
   ```

5. Enable the corresponding CEF events from OpenText Identity Console. For more information, see Configuring the CEF Events for Auditing (https://www.netiq.com/documentation/edirectory-93/edir_admin/data/t44e7j6b8ufi.html#t44e7j6cx14i). Enabled events will be forwarded to the auditing server.

Disaster Recovery

Disaster recovery is performed in case of a VM crash where OpenText eDirectory was running. Perform the following steps for disaster recovery:

1. Stop the VM which has crashed and dissociate the data disk from it. For more information, see How to detach a data disk from a Linux virtual machine.
2. Configure a new VM with the same operating system as the VM which has crashed.
3. Install the same version of OpenText eDirectory in the new VM.
4. Attach the data disk to the new VM and mount the file system. For more information, see Use the portal to attach a data disk to a Linux VM.
5. Bind mount the directories.

   To bind mount the directories, update the following in `/etc/fstab`:

   ```
   <mount_point>/eDirectory_data /var/opt/novell/eDirectory none
   defaults,bind 0 0
   ```

   ```
   <mount_point>/nici_data /var/opt/novell/nici none defaults,bind 0 0
   ```

   ```
   <mount_point>/eDirectory_nici_conf /etc/opt/novell/eDirectory none
   defaults,bind 0 0
   ```

6. Change the IP address in `/etc/opt/novell/eDirectory/conf/nds.conf` to current VM IP address.
7. Upgrade OpenText eDirectory skipping health check. For more information, see *Upgrading OpenText eDirectory* .
8. Repair the network addresses using `ndsrepair` utility. For more information, see *DSRepair Options* in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.
9. Modify the CRL distribution point IP address if the tree CA IP address is changed. For more information on how to change the IP address, see Viewing and Modifying a CRL Configuration Object's Properties (https://www.netiq.com/documentation/edirectory-93/edir_admin/data/b1j4tpo3.html#b1j581ua) in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.
10. Repair the server's default certificates using the OpenText Identity Console certificate management. To repair the default certificates, perform the following steps:

    10a Login to OpenText Identity Console as Administrator.

    10b Go to **Certificates Management** > **Default Certificate**.

    10c Select the server(s) which owns the certificates and click **Next**.

    10d Select **Yes All Default Certificates will be overwritten** and click **Next**.

    10e Review the tasks to be performed and select **Finish** > click **OK**.
11. Configure LDAP and HTTP services to use new ECDSA Certificates.

Deploying on a Windows Platform

The first step in deployment requires you to prepare Azure services for use with OpenText eDirectory. For more information, see “Preparing Azure Services” on page 89. After creating the Azure services, perform the following steps:

- “Configuring Application Security Groups (ASG)” on page 101
- “Configuring Network Security Groups for Virtual Machine” on page 101
- “Creating and Deploying Virtual Machines” on page 104
- “Configuring Data Disk for Storing OpenText eDirectory Data” on page 104
- “Installing OpenText eDirectory and OpenText Identity Console” on page 104
- “Disaster Recovery” on page 107

Configuring Application Security Groups (ASG)

Application security groups enable you to configure network security as a natural extension of an application's structure. It also allows you to group virtual machines and define network security policies based on those groups. For more information, see Application Security Groups.

You must create the following Application security groups before configuring the network security groups:

Table 5-4 *Application Security Groups*

Name	Description
OpenText eDirectory	Contains the network interface of all the virtual machines where OpenText eDirectory will be configured
eDirectory_CA	Contains the network interface of the virtual machine where OpenText eDirectory server hosting the tree CA will be configured
OpenText Identity Console	Contains the network interface of the virtual machine where OpenText Identity Console will be configured

Configuring Network Security Groups for Virtual Machine

A security group is a set of virtual firewall rules which can be assigned to one or more virtual machines in the Virtual Network.

For more information, see Security Groups.

To deploy OpenText eDirectory on Azure, create the following network security groups: eDirectory_NSG_1, eDirectory_NSG_2 and IdentityConsole_NSG. Create these security groups with the following port rules over the default security rules:

1. **eDirectory_NSG_1:** This NSG should be associated with the virtual machine hosting the OpenText eDirectory tree CA.

Priority	Name	Port Range	Source	Destination	Action	Description
100	NCP	TCP 524	OpenText eDirectory subnet	OpenText eDirectory (ASG)	ALLOW	Allows NCP traffic for OpenText eDirectory in OpenText eDirectory subnet
110	HTTP CRL	TCP 8028	Virtual Network	eDirectory_CA (ASG)	ALLOW	Required for accessing the OpenText eDirectory tree CRL from the VNet when there are services in the VNet which are configured with certificates issued by tree CA.
120	LDAPS	TCP 636	OpenText eDirectory subnet	OpenText eDirectory (ASG)	ALLOW	Allows secured LDAP traffic in OpenText eDirectory subnet
130	SLP	Any 427	OpenText eDirectory subnet	OpenText eDirectory (ASG)	ALLOW	Allows SLP traffic in the OpenText eDirectory subnet
140	All Traffic	All	Any	Any	DENY	Denies all inbound traffic

NOTE: ◆OpenText eDirectory subnet refers to the virtual machine where OpenText eDirectory is getting installed.

- OpenText eDirectory servers should not be configured to listen on LDAP port 389 and access to port 389 should not be allowed on the security group which is assigned to OpenText eDirectory. Also access to HTTP port should only be allowed on the security group assigned to OpenText eDirectory server hosting the tree CA.

2. **eDirectory_NSG_2:** This NSG should be associated with all virtual machines hosting OpenText eDirectory servers other than the OpenText eDirectory tree CA.

Priority	Name	Port Range	Source	Destination	Action	Description
100	NCP	TCP 524	OpenText eDirectory subnet	OpenText eDirectory (ASG)	ALLOW	Allows NCP traffic for OpenText eDirectory in backend subnet
110	LDAPS	TCP 636	OpenText eDirectory subnet	OpenText eDirectory (ASG)	ALLOW	Allows secured LDAP traffic in backend subnet
120	SLP	Any 427	OpenText eDirectory subnet	OpenText eDirectory (ASG)	ALLOW	Allows SLP traffic in the backend subnet
130	All Traffic	All	Any	Any	DENY	Denies all inbound traffic

3. **IdentityConsole_NSG:** This NSG should be associated with the virtual machine hosting OpenText Identity Console. The following NSG rules enable access to the OpenText Identity Console server from the Proxy server only.

Priority	Name	Port Range	Source	Destination	Action	Description
100	HTTPS	TCP 9000	Virtual Network - Subnet	OpenText Identity Console (ASG)	ALLOW	Allows secured HTTP traffic for accessing Proxy within the subnet
110	All Traffic	All	Any	Any	DENY	Denies all inbound traffic

Creating and Deploying Virtual Machines

Create and launch your virtual machines (VM) on one of the supported platforms. For more information on how to create and launch VMs, see Create and Launch Your Windows Virtual Machine. As a part of creating and launching the instances, you must also perform the following steps:

1. Associate eDirectory_NSG_1 with the VMs where the first OpenText eDirectory server will be configured, eDirectory_NSG_2 with the VM where all other OpenText eDirectory servers will be configured and IdentityConsole_NSG with the VM where Identity Console will be configured. For more information about security groups, see "Configuring Network Security Groups for Virtual Machine" on page 101.
2. Create Bastion host to connect to windows VM.

   To access Windows VM on Azure, create Bastion. For more details on how to configure Bastion and connect to a Windows VM, see Configuring Bastion.

Configuring Data Disk for Storing OpenText eDirectory Data

Configuring data disk is required to prevent loss of OpenText eDirectory data and configuration in case of Azure vm crash. For more information on recovering OpenText eDirectory data and configuration, see "Disaster Recovery" on page 107. After creating the VM, perform the following steps to prepare the VM for deploying OpenText eDirectory:

1. Create and attach a data disk, perform the steps in Use the portal to attach a data disk to a Windows VM.
2. Login to the VM, format the data disk with NTFS file system and mount the data disk. For more information on how to format and mount the data disk, see Initialize a new data disk.

Installing OpenText eDirectory and OpenText Identity Console

Prerequisites

- ❒ Ensure that you meet the requirements listed in "System Requirements" on page 67.
- ❒ Create security groups as mentioned in "Configuring Network Security Groups for Virtual Machine" on page 101.
- ❒ Choose a VM size that provides good performance and memory for this instance.
- ❒ Configure data disk for storing OpenText eDirectory data.
- ❒ Configure Bastion to connect to Windows VM.

Installation and Configuration Procedure

This section explains the step by step instructions to install and configureOpenText eDirectory and OpenText Identity Console in an Azure environment. Once OpenText eDirectory is installed, you should ensure that the following conditions are met:

- EBA is enabled
- SNMP is disabled
- OpenText eDirectory is not listening on port 389

- LDAP and HTTP services are configured to use ECDSA certificates only
- Access to the RDP port of the Azure VMs subnet should be disabled when not in use
- Disable OpenText iMonitor, eMBox and DHost modules to provide additional security. After disabling them, all activities involving these modules should be performed using NDS utilities only.

Installing & Configuring OpenText eDirectory

1. Download the `eDirectory_<version>_Windows_x86_64.exe` file.
2. Install eDirectory. For more information, see Installing or Upgrading OpenText eDirectory CE 24.4 (v9.3) on Windows.

Installing & Configuring OpenText Identity Console

Using the OpenText Identity Console administrative console, you can manage the OpenText eDirectory operations on your Azure environment. Identity Console should be installed on your Azure VM after installing OpenText eDirectory.

1. Download the `IdentityConsole_<version>_win_x86_64.zip` file.
2. Install and configure OpenText Identity Console. For more information, see Installing OpenText Identity Console on Windows (https://www.netiq.com/documentation/identity-console/identity_console-install/data/t4j68b4oo7x1.html). Before installing OpenText Identity Console, see the system requirements in System Requirements (https://www.netiq.com/documentation/identity-console/identity_console-install/data/t4ja6mu8vquv.html) section in the OpenText Identity Console Installation Guide. (https://www.netiq.com/documentation/identity-console/identity_console-install/data/bookinfo.html)

Launching OpenText Identity Console

Perform the following steps to launch OpenText Identity Console:

1. Open the Bastion host user interface.
2. Launch OpenText Identity Console in a new browser tab.
3. Enter the OpenText Identity Console Port and URL.

Post-Configuration Tasks

1. To check if EBA is enabled, see Viewing Information About EBA in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
2. Enable Suite B on Certificate Server. For more information, see Enabling Suite B on the Certificate Server in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
3. Configure AES 256-bit tree key for the first OpenText eDirectory server. For more information, see Creating an AES 256-Bit Tree Key in the *NICI Administration Guide*.
4. Delete the CRL distribution points in the first OpenText eDirectory server. As non-secured LDAP access over port 389 is disabled on all OpenText eDirectory servers, the CRL for the tree CA should be available for download over HTTP only. Perform the following steps to delete the CRL distribution points:
   - **4a** Login to OpenText Identity Console as Administrator.
   - **4b** Go to **Certificate Management** > **CA Management** > **CRL** > Create CRL .

4c Click **One**. Select and delete all **CRL Distribution Points** except the HTTP CRL Distribution Point (http://<instance_ip>:8028/crl/one.crl).

4d Click **Save** and then click **OK**.

4e Click **OneEC**. Select and delete all **CRL Distribution Points** except the HTTP CRL Distribution Point (http://<instance_ip>:8028/crl/oneec.crl).

4f Click **Save** and then click **OK**.

5 Repair the server's default certificates using the OpenText Identity Console certificate management. To repair the default certificates, perform the following steps:

5a Login to OpenText Identity Console as Administrator.

5b Go to **Certificates Management** > **Default Certificate**.

5c Select the server(s) which owns the certificates and click **Next**.

5d Select **Yes All Default Certificates will be overwritten** and click **Next**.

5e Review the tasks to be performed and select **Finish** > click **OK**.

6 Configure LDAP and HTTP services to use ECDSA Certificates and Suite B ciphers. For more information, see Configuring LDAP and HTTP Services to Use ECDSA Certificates and Suite B Ciphers in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*. Once done, restart OpenText eDirectory.

7 For more information to check if SNMP sub-agent is unloaded, see Loading and Unloading the SNMP Server Module in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

8 Ensure that OpenText eDirectory is not listening on port 389.

9 Disable OpenText iMonitor, eMBox, DHost and HTTP stack.

9a Perform the following steps to disable OpenText iMonitor, eMBox and DHost in the OpenText eDirectory server hosting the tree CA:

9a1 Edit the `ndsmodules.conf` file by commenting `hconserv`, `imon` and `embox`.

9a2 Restart OpenText eDirectory.

9b Perform the following steps to disable the HTTP stack in the OpenText eDirectory replica servers:

9b1 Edit the `ndsmodules.conf` file by commenting `httpstk`, `hconserv`, `imon` and `embox`.

9b2 Restart OpenText eDirectory.

NOTE: `httpstk` should be placed above `nds` in the `ndsmodules.conf` file before commenting. This stops `nds` module from enabling HTTP stack.

10 Configure SLP to force OpenText eDirectory to use unicast as advertising method. Edit the `slp.conf` file by providing the IP address of the DA server in the Backend subnet. For more information, see Configuration Parameters in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

NOTE: Once all OpenText eDirectory VMs and Identity Console have been configured, configure the security rules of the Azure backend subnet to deny access to the SSH port and allow it only when required.

Disaster Recovery

Perform the following steps for disaster recovery on a Windows virtual machine:

NOTE: It is recommended to take backup of the nici files using dsbk backup after configuring a tree. For more information, see Disaster Recovery Plan on Windows.

---

1. Stop the VM which has crashed and dissociate the data disk from it. For more information, see How to detach a data disk from a Windows virtual machine.
2. Configure a new VM with the same operating system as the VM which has crashed.
3. Install the same version of OpenText eDirectory in the new VM.
4. Copy the dsbk backup files with nici backup to the new VM. Stop the ndsd services and run dsbk restore with nici.
5. Attach the data disk to the new VM and mount the file system. For more information, see Use the portal to attach a data disk to a Windows VM.
6. Start ndsd services on the new VM.
7. Upgrade OpenText eDirectory. For more information, see *Upgrading OpenText eDirectory* in the OpenText™ eDirectory CE 24.4 (v9.3) Installation Guide.
8. Repair the network addresses using `ndsrepair` utility. For more information, see DSRepair Options in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.
9. Modify the CRL distribution point IP address if the tree CA IP address is changed. For more information on how to change the IP address, see Viewing and Modifying a CRL Configuration Object's Properties (https://www.netiq.com/documentation/edirectory-93/edir_admin/data/b1j4tpo3.html#b1j581ua) in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.
10. Repair the server's default certificates using the OpenText Identity Console certificate management. To repair the default certificates, perform the following steps:
    - **10a** Login to OpenText Identity Console as Administrator.
    - **10b** Go to **Certificates Management** > **Default Certificate**.
    - **10c** Select the server(s) which owns the certificates and click **Next**.
    - **10d** Select **Yes All Default Certificates will be overwritten** and click **Next**.
    - **10e** Review the tasks to be performed and select **Finish** > click **OK**.
11. Configure LDAP and HTTP services to use new ECDSA Certificates.

Deploying OpenText eDirectory Container on Microsoft Azure Container Instance

OpenText eDirectory supports OpenText eDirectory container deployment on Microsoft Azure container instance. The deployment process is simple and time-efficient. OpenText eDirectory container is pushed into the registry as Docker image that is self-contained and capable of running on its own. This section guides you through the process of deploying OpenText eDirectory Docker image on Azure.

Checklist for Deploying the Container

To setup the Azure container for deployment, we recommend that you complete the steps in the following checklist:

	Checklist Items
❑	1. You must create an Azure container registry to store and manage the OpenText eDirectory Docker container image. Sign in to the Azure portal at https://portal.azure.com/ and follow the steps in the Create an Azure Container Registry Using the Azure Portal process to create a container registry.
❑	2. Next, you must set up an Azure storage account. This account stores all your data which can be accessed from anywhere in the world via HTTP or HTTPS. Sign in to the Azure portal and follow the steps in the Create a Storage Account process to create an Azure storage account.
❑	3. After you have created the storage account, create an Azure file share. A shared file system is required to allow the containers to access the file system regardless of which instance they run on. For more information on the considerations and process to follow when creating an Azure file share, see Create an Azure file share.
❑	4. You must create a virtual network. Azure Virtual Network enables many types of Azure resources, such as Azure Virtual Machines (VM), to securely communicate with each other within the Azure Internet and on-premises networks. Follow the steps in the Creating a Virtual Network process.
❑	5. Then, push the OpenText eDirectory Docker image to the container registry. For more information, see “Pushing Docker Image to Azure Container Registry” on page 108.
❑	6. Edit and save the YAML file bundled with the OpenText eDirectory Docker image tarball. For more information, see “Editing the YAML Configuration File” on page 109.
❑	7. Finally, create an Azure container instance using the YAML file. For more information, see “Creating an Azure Container Instance” on page 113.

Pushing Docker Image to Azure Container Registry

To push the Docker image to the Azure container registry:

1. Download the OpenText eDirectory Docker image tarball from the Software License and Download portal.
2. Untar the downloaded build, extract the Docker image, and then load it into the local Docker registry using the following commands:

```
tar -xvf eDirectory_93x_Container.tar.gz
docker load --input eDirectory_93x/eDirectory_93x.tar.gz
```

The docker load command loads the Docker image named `edirectory:9.3.x` from tar archive.

3 Tag the Docker image with your registry name using the following command

```
Docker tag edirectory:<image-id> <registry-name>.azurecr.io/
<repository-name-tag>
```

Where,

`edirectory:<image-id>` is the local Docker image ID.

`<registry-name>` is the name of your private registry.

`<repository-name-tag>` is the tag you want to assign to the Docker image.

For example, `docker tag edirectory:93x myazureregistry.azurecr.io/edir93`

4 Push the Docker image to the registry using the following command:

```
docker push <registry-name>.azurecr.io/<repository-name-tag>
```

Where,

`<registry-name>` is the name of your private registry.

`<repository-name-tag>` is the tag that you assigned to the Docker image.

For example, `docker push myazureregistry.azurecr.io/edir93x`

Editing the YAML Configuration File

A YAML file is a quick and easy way to create or update a container group. The `az container create` command in the Azure command-line interface reads the group configurations from a YAML file. You must edit and upload the YAML file bundled with the OpenText eDirectory Docker image tarball before creating an Azure container instance.

The following section shows the schema for a YAML file, including comments (indicated with a # symbol) that highlights key properties. For a description of properties in this schema, see the Property Values section.

```
name:  # Provide the name of the container group
apiVersion: '2019-12-01'
properties: # Properties of container group
  containers: # The containers within the container group
  - name: test-edir93 # Provide a name of container instance
    properties: # Properties of a container instance
      image: myazureregistry.azurecr.io/edir93 # Provide the name of the
container image used to create the container instance
      command:
      - /start.sh #Not optional
      - 'new -t 93edir1 -n novell -a admin.novell -w xx -S server1 -i -B
@1524 -o 1028 -O 1030 -L 1389 -l 1636 --configure-eba-now no' #Sample
command
      ports: # External-facing ports exposed on the instance, must also be
set in group ipAddress property
      - protocol: tcp
       port: 1030 #HTTPS port. Depends on the parameter provided in command
```

```
for -O
      - protocol: tcp
        port: 1389 #LDAP port. Depends on the parameter provided in command
for -L
      environmentVariables:
      - name: YAML_MODE #Mandatory variable. Required if yaml file is used
for configuring edirectory
        value: '1'
      resources: # Minimum resource requirements of the instance
        requests:
          memoryInGB: 2
          cpu: 1.5
      volumeMounts: # Array of volume mounts for the instance
      - name: configvolume
        mountPath: /config #Mandatory. Do not change this path.
  imageRegistryCredentials: # Credentials to pull a private image
  - server: myazureregistry.azurecr.io #Provide the image registry server
name
    username: myazureregistry #Provide the image registry username
password: xxxx #Provide the image registry password
  #restartPolicy: Never #Optional
  ipAddress: # IP address configuration of container group
    ports:
    - protocol: tcp
      port: 1030 #HTTPS port. Depends on the parameter provided in command
for -O
    - protocol: tcp
      port: 1389 #LDAP port. Depends on the parameter provided in command
for -L
    type: Private
  networkProfile: # Virtual network profile for container group
    id: subscriptions/xxx-xxx-xxx/resourceGroups/myresourcegroup/
providers/Microsoft.Network/networkProfiles/aci-network-profile-
VirtualNetmy-containersubnet
#Can be found by running the command, az network profile list --resource-
group myResourceGroup --query [0].id --output tsv, if no output then run
the first time using azure cloud shell
  osType: Linux
  volumes: # Array of volumes available to the instances
  - name: configvolume
    azureFile:
      shareName: testshare  #Provide the name of the Azure File share to be
mounted as a volume
      storageAccountName: myfileshare  #Provide name of the storage account
      storageAccountKey: xxx_xxx #Provide storage account access key
```

Property Values

The following tables describe the values you need to set in the schema.

Name	Type	Required	Value
containerGroups	**object**		

Name	Type	Required	Value
Name	String	Yes	Name of the container group. You can also provide container name.
apiVersion	enum	Yes	The resource api version. For example, '2019-12-01'.
properties	object	Yes	Properties of a container group. Look for the `ContainerGroupProperties` object in this table to find values supported for properties.
`ContainerGroupProperties` object			
containers	array	Yes	The containers within the container group. Look for the `Container` object in this table to find values supported for containers.
imageRegistryCredentials	array	No	The image registry credentials by which the container group is created from. Look for the `ImageRegistryCredential` object in this table to find values supported for imageRegistryCredentials.
restartPolicy	enum	No	Restart policy for all containers within the container group: ◆ - Always: Always restart ◆ - OnFailure: Restart on failure ◆ - Never: Never restart
ipAddress	object	No	The IP address type of the container group. Look for the `IPAddress` object in this table to find values supported for ipAddress.
osType	enum	Yes	The operating system type required by the containers in the container group. - Windows or Linux
volumes	array	No	The list of volumes that can be mounted by containers in this container group. Look for the `Volume` object in this table to find values supported for volumes.
`Container` object			
name	string	Yes	The user-provided name of the container instance. For example, `test-edir93`.
properties	object	Yes	The properties of the container instance. Look for the `ContainerProperties` object in this table to find values supported for properties.
`ContainerProperties` object			
image	string	Yes	The name of the image used to create the container instance. For example, `myazureregistry.azurecr.io/edir93` (Step 3).

Name	Type	Required	Value
command	array	No	The commands to execute within the container instance in exec form. OpenText eDirectory Docker container accepts the parameters of the `ndsconfig` utility with Docker command. For more information about the `ndsconfig` utility, see "The ndsconfig Utility" on page 157.
ports	array	No	The exposed ports on the container instance. Look for the `ContainerPort` object in this table to find values supported for ports.
environmentVariables	array	No	The environment variables to set in the container instance. Look for the `EnvironmentVariable` object in this table to find values supported for environmentVariables.
resources	object	Yes	The resource requirements of the container instance. Look for the `ResourceRequirements` object in this table to find values supported for resources.
volumeMounts	array	No	The volume mounts available to the container instance. Look for the `VolumeMount` object in this table to find values supported for volumeMounts.
`ContainerPort` object			
protocol	enum	No	The protocol associated with the port. - TCP or UDP
port	integer	Yes	The port number.
`EnvironmentVariable` object			
name	string	Yes	The name of the environment variable.
value	string	No	The value of the environment variable.
`ResourceRequirements` object			
requests	object	Yes	The resource requests of this container instance. Look for the `ResourceRequests` object in this table to find values supported for requests.
`ResourceRequests` object			
memoryInGB	number	Yes	The memory request in GB of this container instance.
cpu	number	Yes	The CPU request of this container instance.
`VolumeMount` object			
name	string	Yes	The name of the volume mount.
mountPath	string	Yes	The path within the container where the volume should be mounted. Must not contain colon (:).
`ImageRegistryCredential` object			

Name	Type	Required	Value
server	string	Yes	The Docker image registry server without a protocol such as "http" and "https". Provide the registry name created in checklist step 1 (see Checklist for Deploying the Container).
username	string	No	The username for the private registry. For more information, see How to Find Username Information.
password	string	No	The password for the private registry. For more information, see How to Find Azure Registry Password.
`IPAddress` object			
ports	array	Yes	The list of ports exposed on the container group. Look for the `Port` object in this table to find values supported for ports.
type	enum	Yes	Specifies if the IP is exposed to the public internet or private VNET. - Public or Private
`Port` object			
protocol	enum	No	The protocol associated with the port. - TCP or UDP
port	integer	Yes	The port number.
`Volume` object			
name	string	Yes	The name of the volume.
azureFile	object	No	The Azure File volume. Look for the `AzureFileVolume` object in this table to find values supported for azureFile.
`AzureFileVolume` object			
shareName	string	Yes	The name of the Azure File share to be mounted as a volume. Provide the storage file share name created in checklist step 3 (see Checklist for Deploying the Container).
storageAccountName	string	Yes	The name of the storage account that contains the Azure File share.
storageAccountKey	string	No	The storage account access key used to access the Azure File share. To find storage account key, see View Account Access Keys.

Creating an Azure Container Instance

Run the following command to create a container in a container group using the YAML file you earlier edited:

```
az container create --resource-group <MyResourceGroup> --name <container-instance-name>  --file <containerGroup.yaml>
```

Where,

<MyResourceGroup> represents the name of the resource group. Mandatory field.

<container-instance-name> represents the name of the container instance in a container group.

<containerGroup.yaml> represents the path to the input file.

For example, `az container create  --resource-group MyResourcegroup --name edir93  --file edir-azure-conf.yaml`

6 Deploying OpenText eDirectory on Amazon Web Services EC2

OpenText eDirectory can be deployed on Amazon Web Services (AWS) EC2 instances. For more information on the supported operating systems, see “System Requirements” on page 27.

The following sections explain various prerequisites and procedure for deploying OpenText eDirectory on AWS:

- “Prerequisites” on page 115
- “Deployment Procedure” on page 115
- “Sizing Guidance for OpenText eDirectory Deployment on AWS” on page 127

Prerequisites

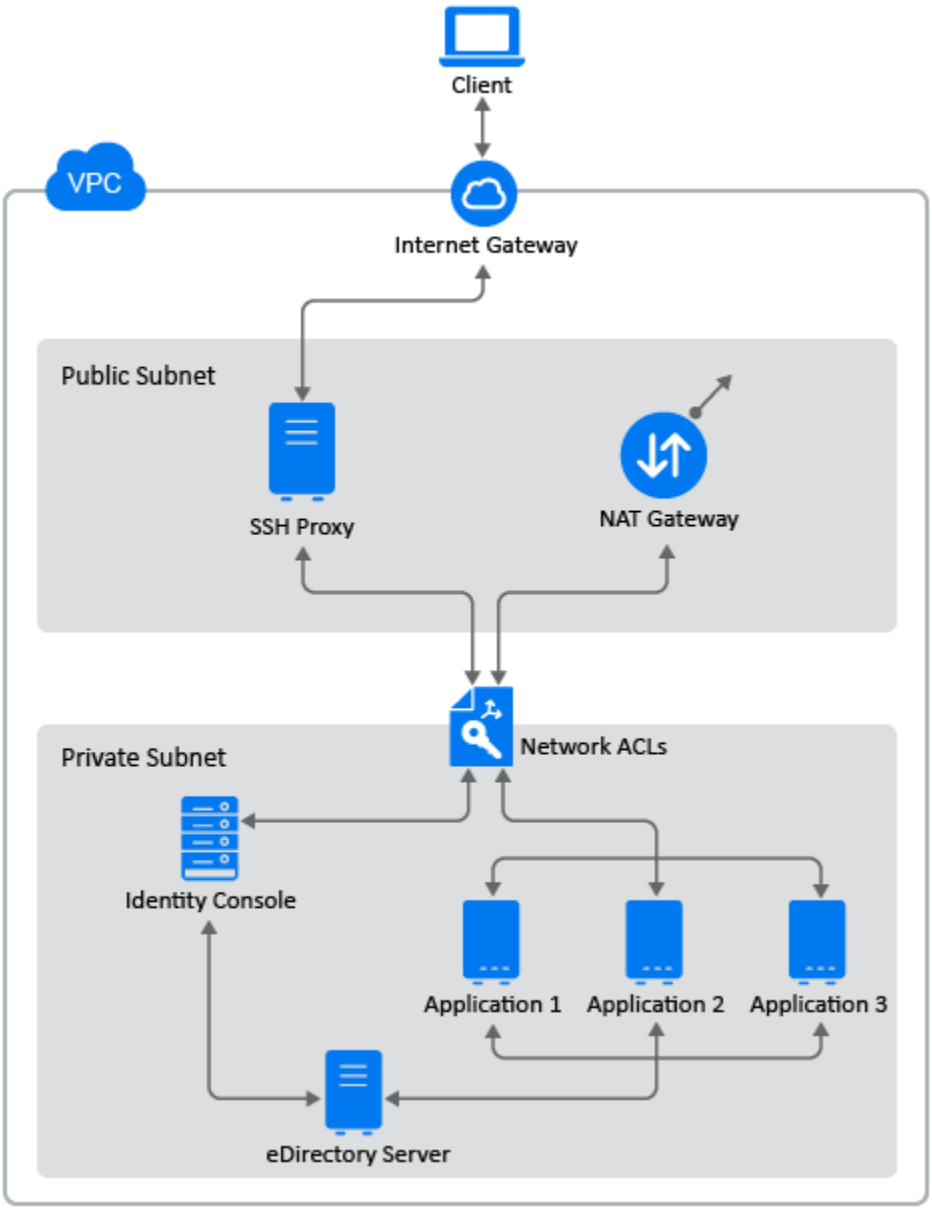
In addition to the system requirements of OpenText eDirectory, ensure that you meet the following requirements:

- An administrative account on AWS EC2.
- The OpenText eDirectory installer (tarball) has been downloaded, extracted, and available for copying to the instances.
- An SSH client to connect to the AWS EC2 instances from the client machine.

Deployment Procedure

OpenText eDirectory should be deployed only in a private subnet in Amazon VPC. Figure 5-1 illustrates a sample deployment that is used in the subsequent sections.

Figure 6-1 *OpenText eDirectory Deployment on AWS EC2*



NOTE: ◆SSH Proxy is bastion host in the public subnet to which the administrator connects using SSH and connects to other instances in the private subnet using the SSH agent forwarding.

- Applications that need to access OpenText eDirectory should be deployed in the private subnet. If these applications need to be accessed from the Internet, configure an AWS EC2 load balancer in the public subnet to enable the access. For more information, see Create an Application Load Balancer.

The deployment procedure consists of the following steps:

- "Preparing AWS Virtual Private Cloud" on page 117
- "Configuring Network ACLs" on page 118
- "Configuring Security Groups" on page 120

- "Creating a SSH Key Pair" on page 121
- "Creating and Deploying Instances" on page 121
- "Configuring EBS Volume for Storing OpenText eDirectory Data" on page 121
- "Installing OpenText eDirectory and OpenText Identity Console" on page 122
- "Deploying Auditing Services" on page 126
- "Disaster Recovery" on page 126

Preparing AWS Virtual Private Cloud

This section outlines general steps to configure AWS VPC to use with OpenText eDirectory. For more information, see the Amazon Elastic Compute Cloud Documentation.

Perform the following steps to create AWS VPC services:

1 Log in to the AWS Management Console.

2 Create the following services:

Service	Description
VPC	You can create a VPC using the Amazon VPC console. For more information on how to create a VPC, see Creating a VPC. For more information on the overview of VPC, see the Amazon Virtual Private Cloud Documentation.
IMPORTANT: Creating a VPC using **Start VPC Wizard** creates two Subnets, Internet gateways, and Route table and NAT gateway for the VPC. You can view or edit these items as follows:	
Subnets	As part of the VPC creation, two subnets will be created; a public and a private subnet. OpenText eDirectory should be deployed in the private subnet. As shown in Figure 5-1, any application which is accessing OpenText eDirectory, should be deployed in the same private subnet. SSH access to the instances in the private subnet should be done via SSH proxy in the public subnet. For more information, see VPC and Subnet. (https://docs.aws.amazon.com/vpc/latest/userguide/configure-subnets.html)
Internet gateways	Internet gateway is required to enable SSH connection to SSH proxy as shown in Figure 5-1. For more information to create and attach Internet gateways with VPC, see Internet Gateways.
Route table	For more information to create a route table, see Route Tables.
NAT gateway	NAT gateway is required for those instances which are in the private subnet to download operating system updates. For more information to create a NAT gateway, see NAT Gateways.
Elastic IP address	Elastic IP addresses are public static IP addresses which should be assigned to the instance running SSH proxy and NAT gateway. For more information to create Elastic IP address, see Associate Elastic IP addresses with resources in your VPC (https://docs.aws.amazon.com/vpc/latest/userguide/vpc-eips.html#WorkWithEIPs)

Configuring Network ACLs

A network access control list (ACL) is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets. For more information, see Network ACLs.

This section outlines the rules to create network ACLs in the private subnet. Configure the following rules for network ACLs in the private subnet:

- Inbound Rules:

Table 6-1

Rule	Type	Port Range	Source	Action	Description
10	SSH	TCP 22	*<SSH Proxy IP Address>*/32	ALLOW	Allows inbound SSH traffic from your SSH proxy IP address in public subnet
15	HTTPS	TCP 9000	*<SSH Proxy IP Address>*/32	ALLOW	Allows inbound HTTPS traffic from the SSH proxy IP address in public subnet
20	Custom TCP Rule	TCP 32768-65535	0.0.0.0/0	ALLOW	Allows inbound return traffic from hosts on the Internet that are responding to requests originating in the subnet
*	All Traffic	All	0.0.0.0/0	DENY	Denies all inbound IPv4 traffic not already handled by a preceding rule (not modifiable)

- Outbound Rules:

Table 6-2

Rule	Type	Port Range	Destination	Action	Description
10	Custom TCP Rule	TCP 32768-65535	*<SSH Proxy IP Address>*/32	ALLOW	Allows outbound SSH traffic from the private subnet to SSH proxy in the public subnet
12	Custom TCP Rule	TCP 32768-65535	*<SSH Proxy IP Address>*/32	ALLOW	Allows outbound traffic for accessing Identity Console from the public subnet
15	HTTPS	TCP 443	0.0.0.0/0	ALLOW	Allows outbound HTTPS traffic from the private subnet to the Internet
20	HTTP	TCP 80	0.0.0.0/0	ALLOW	Allows outbound HTTP traffic from the private subnet to the Internet
25	HTTPS	Port number to which the auditing server is listening	*<IP Address of the audit server>*	ALLOW	Allow auditing eDirectory events. **NOTE:** This rule is applicable only when the audit server is outside the private subnet.
*	All Traffic	All	0.0.0.0/0	DENY	Denies all outbound IPv4 traffic not already handled by a preceding rule (not modifiable)

Configuring Security Groups

A security group is a set of virtual firewall rules which can be assigned to one or more instances in the VPC.

By default, a new security group only allows incoming traffic on port 22, so that you can only connect to the instance by using SSH.

For more information, see Amazon EC2 Security Groups for Linux Instances.

To deploy OpenText eDirectory on AWS, create three security groups. For example Security Group 1, Security Group 2 and Security Group 3. Create these security groups with the following port rules:

1. **Security Group 1 (for eDirectory):**

Port	Source	Description
TCP 22	Public subnet	Allows SSH traffic from public subnet
TCP 636	Private subnet	Allows LDAPS traffic in private subnet
TCP 524	Private subnet	Allows NCP traffic for eDirectory in private subnet
UDP 427	Private Subnet	Allows SLP traffic in the private subnet

NOTE: LDAP port 389 should not be enabled on the security group which is assigned to OpenText eDirectory. HTTP port should only be enabled on the security group assigned to OpenText eDirectory server hosting the tree CA.

2. **Security Group 2 (for eDirectory):**

Port	Source	Description
TCP 8028	VPC	Required for accessing the OpenTexr eDirectory tree CRL from the VPC when there are services in the VPC which are configured with certificates issued by tree CA. This security group is assigned to the OpenTexreDirectory server hosting t he tree CA.

3. **Security Group 3 (for Identity Console):**

Port	Source	Description
TCP 22	Public subnet	Allows SSH traffic from public subnet
TCP 9000	Public subnet	Allows HTTPS traffic for accessing Identity Console from public subnet

Creating a SSH Key Pair

You must create a SSH key pair before configuring the Amazon EC2 instances. To create a key pair, perform the following steps:

1. Create a 4096-bit RSA SSH key pair on your client using the following command:

   ```
   ssh-keygen -t rsa -b 4096
   ```

   `ssh-keygen` places the newly created public key at `~/.ssh/id_rsa.pub`.

2. Import the SSH public key to your Amazon EC2 account. For more information, see Importing Your Own Public Key to Amazon EC2 (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/create-key-pairs.html).

IMPORTANT: You can connect to and manage your instances only using the SSH private key. Therefore, do not lose the SSH private key.

Creating and Deploying Instances

Create and launch your EC2 instances on one of the supported platforms. For more information on how to create and launch instances, see Create Your EC2 Resources and Launch Your EC2 Instance. As a part of creating and launching the instances, you must also perform the following steps:

1. Associate Security Group 1 with the instances where first OpenText eDirectory server will be configured, Security Group 2 with the instance where all other OpenText eDirectory servers will be configured and Security Group 3 with the instance where Identity Console will be configured. For more information about security groups, see "Configuring Network Security Groups for Virtual Machine" on page 92.
2. Associate the public key created in section "Creating a SSH Key Pair" on page 94 with your instances.

Configuring EBS Volume for Storing OpenText eDirectory Data

Configuring EBS volume is required to prevent loss of OpenText eDirectory data and configuration in case of EC2 instance crash. For more information on recovering OpenText eDirectory data and configuration, see "Disaster Recovery" on page 126. After creating the EC2 instance, perform the following steps to prepare the instance for deploying OpenText eDirectory:

1. Create an EBS volume, perform the steps in Creating an Amazon EBS Volume.
2. Associate the EBS volume with the EC2 instance. For more information, see Attaching an Amazon EBS Volume to an Instance
3. Login to the instance, format the EBS volume with `ext4` file system and mount the EBS volume. For more information on how to format and mount the EBS volume, see Making an Amazon EBS Volume Available for Use on Linux.
4. Bind mount directories from the EBS volume to OpenText eDirectory data/NICI data directories. Perform the following steps as root user to bind mount:

   4a Create OpenText eDirectory data directory by using the following command:

   ```
   mkdir <mount_point>/eDirectory_data
   ```

4b Create NICI data directory by using the following command:

```
mkdir <mount_point>/nici_data
```

4c Create NICI and OpenText eDirectory configuration directories by using the following command:

```
mkdir <mount_point>/eDirectory_nici_conf
```

4d Create required directories for OpenText eDirectory by using the following commands:

```
mkdir –-parents /var/opt/novell/eDirectory
mkdir –-parents /var/opt/novell/nici
mkdir –-parents /etc/opt/novell/eDirectory
```

4e To bind mount the directories, add the following to `/etc/fstab`:

```
<mount_point>/eDirectory_data /var/opt/novell/eDirectory none
defaults,bind 0 0

<mount_point>/nici_data /var/opt/novell/nici none defaults,bind 0 0

<mount_point>/eDirectory_nici_conf /etc/opt/novell/eDirectory none
defaults,bind 0 0
```

NOTE: All operations in the instance should be performed as a root user.

Installing OpenText eDirectory and OpenText Identity Console

Prerequisites

- ❒ Ensure that you meet the requirements listed in System Requirements.
- ❒ Create security groups as mentioned in “Configuring Network Security Groups for Virtual Machine” on page 92.
- ❒ SSH proxy instance should be hardened and secured server. Open only SSH port 22 for this instance and select an AWS instance type with good performance and memory. SSH private key required for accessing instances in the private subnet and the instance where SSH proxy is running, should not be stored in the VPC but on the client only. Associate an Elastic IP address to the SSH Proxy server to have static public IP address.
- ❒ Configure the VNC server in the SSH Proxy instance. VNC server should be hardened with a password of good strength. Connect to the VNC server through an SSH tunnel only to allow a secured communication. VNC server should be configured to listen only for connections from the localhost. Disable screen lock to avoid session lockout. After using the VNC server, you should terminate the session.
- ❒ Connect to the instance in the private subnet where OpenText eDirectory/OpenText Identity Console will be configured using SSH proxy:

  ```
  ssh -i edir_key.pem -A -J ec2-user@<ssh_proxy_ip> ec2-
  user@<instance_private_ip>
  ```

NOTE: ◆in above sample commands, `edir_key.pem` is a sample file name containing the server key.

- You can also add the identity file in the agent using the SSH-Add command to avoid using identity file every time you login.

To view the private IP address of an instance, click **Instances** > *[instance]* > **Description**.

❒ Configure an SLP Directory Agent (DA) server in a VM in the Backend subnet. Open port 427 in the inbound rule of NSG for the VM where SLP DA is deployed. Enable DA operation by editing the `slp.conf` file. For more information, see Configuring OpenSLP for OpenText eDirectory in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Installation and Configuration Procedure

This section explains the step by step instructions to install and configure OpenText eDirectory and OpenText Identity Console in an AWS EC2 environment. Once OpenText eDirectory is installed, you should ensure that the following conditions are met:

- EBA is enabled
- SNMP is disabled
- OpenText eDirectory is not listening on port 389
- LDAP and HTTP services are configured to use ECDSA certificates only
- Access to the SSH port of the AWS EC2 private instance should be disabled when not in use.
- Disable OpenText iMonitor, eMBox and DHost modules to provide additional security. After disabling them, all activities involving these modules should be performed using the NDS utilities only.

Installing & Configuring OpenText eDirectory

1. Copy the `eDirectory_<version>_Linux_x86_64.tar.gz` file using Secure Copy (scp) to the instance in the private subnet where OpenText eDirectory will be configured using SSH proxy:

   ```
   scp -i <keyname> -o ProxyJump=ec2-user@<ssh_proxy_ip>
   eDirectory_<version>_Linux_x86_64.tar.gz ec2-user@<instance_ip>:/
   <directory>
   ```

2. Install OpenText eDirectory. For more information, see Using the nds-install Utility to Install OpenText eDirectory Components.
3. Configure OpenText eDirectory. For more information, see Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server. For example, here's a sample command for installing and configuring OpenText eDirectory:

   ```
   ndsconfig  new [-t <tree_name>] [-n <server context>] -a <admin FDN> [-
   w <admin password>] -P ldaps://<instance_ip> --configure-eba-now yes
   ```

4. Install `openslp-server` and start the SLPD service.

Installing & Configuring OpenText Identity Console

Using the OpenText Identity Console administrative console, you can manage the OpenText eDirectory operations on your AWS environment. OpenText Identity Console should be installed on your AWS instance after installing OpenText eDirectory.

1. Copy the `IdentityConsole_<version>_linux_x86_64.tgz` file using Secure Copy (scp) to the instance in the private subnet where OpenText Identity Console will be configured using SSH proxy:

   ```
   scp -i <keyname> -o ProxyJump=ec2-user@<ssh_proxy_ip>
   IdentityConsole_<version>_linux_x86_64.tgz ec2-user@<instance_ip>:/
   <directory>
   ```

2. Install and configure OpenText Identity Console. For more information, see Installing Standalone OpenText Identity Console (Non-Docker). Before installing OpenText Identity Console, see the system requirements in System Requirements and Prerequisites for Standalone Installation (Non-Docker) section in the OpenText™ Identity Console Installation Guide.
3. Replace the Self-Signed certificates in the VM running Identity Console with a secure CA signed certificates. For more information, see Replacing the Temporary Self-Signed Certificates for iManager.

NOTE: Ensure to configure the OpenText Identity Console server to use ECDSA certificates only. After installing OpenText Identity Console, specify an authorized user and the appropriate OpenText eDirectory tree name that this user will manage.

Launching OpenText Identity Console

Perform the following steps to launch OpenText Identity Console:

1. Connect to the VNC server running on localhost of the SSH Proxy through SSH tunnel.
2. Install and launch a browser in the same instance.
3. Launch OpenText Identity Console and connect to the OpenText eDirectory tree using the IP address or the tree name.

Post-Configuration Tasks

1. To check if EBA is enabled, see Viewing Information About EBA in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
2. Enable Suite B on Certificate Server. For more information, see Enabling Suite B on the Certificate Server in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
3. Configure AES 256-bit tree key for the first OpenText eDirectory server. For more information, see Creating an AES 256-Bit Tree Key in the *NICI Administration Guide*.

4 Delete the CRL distribution points in the first OpenText eDirectory server. As non-secured LDAP access over port 389 is disabled on all OpenText eDirectory servers, the CRL for the tree CA should be available for download over HTTP only. Perform the following steps to delete the CRL distribution points:

4a Login to OpenText Identity Console as Administrator.

4b Go to **Certificate Management** > **CA Management** > **CRL** > Create CRL .

4c Click **One**. Select and delete all **CRL Distribution Points** except the HTTP CRL Distribution Point (http://<instance_ip>:8028/crl/one.crl).

4d Click **Save** and then click **OK**.

4e Click **OneEC**. Select and delete all **CRL Distribution Points** except the HTTP CRL Distribution Point (http://<instance_ip>:8028/crl/oneec.crl).

4f Click **Save** and then click **OK**.

5 Repair the server's default certificates using the OpenText Identity Console certificate Manager. To repair the default certificates, perform the following steps:

5a Login to OpenText Identity Console as Administrator.

5b Go to **Certificates Management** > **Default Certificate**.

5c Select the server(s) which owns the certificates and click **Next**.

5d Select **Yes All Default Certificates will be overwritten** and click **Next**.

5e Review the tasks to be performed and select **Finish** > click **OK**.

6 Configure LDAP and HTTP services to use ECDSA Certificates and Suite B ciphers. For more information, see Configuring LDAP and HTTP Services to Use ECDSA Certificates and Suite B Ciphers in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*. Once done, restart OpenText eDirectory.

7 For more information to check if SNMP sub-agent is unloaded, see Loading and Unloading the SNMP Server Module in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

8 Ensure that OpenText eDirectory is not listening on port 389.

9 Disable OpenText iMonitor, embox, DHost and HTTP stack.

9a Perform the following steps to disable OpenText iMonitor, embox and DHost in the eDirectory server hosting the tree CA:

9a1 Edit the `ndsmodules.conf` file by commenting `hconserv`, `imon` and `embox`.

9a2 Restart OpenText eDirectory.

9b Perform the following steps to disable the HTTP stack in the eDirectory replica servers:

9b1 Edit the `ndsmodules.conf` file by commenting `httpstk`, `hconserv`, `imon` and `embox`.

9b2 Restart OpenText eDirectory.

NOTE: `httpstk` should be placed above `nds` in the `ndsmodules.conf` file before commenting. This stops `nds` module from enabling HTTP stack.

---

10 Configure SLP to force OpenText eDirectory to use unicast as advertising method. Edit the `slp.conf` file by providing the IP address of the DA server in the Backend subnet. For more information, see Configuration Parameters in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

NOTE: Once all OpenText eDirectory instances and OpenText Identity Console have been configured, configure the network ACL of the AWS private subnet to deny access to the SSH port and allow it only when required.

Deploying Auditing Services

You can deploy the Common Event Format (CEF) auditing service on AWS EC2 to audit various OpenText eDirectory events. Perform the following steps to deploy CEF auditing services:

1. Install an auditing server in the VPC.
2. Configure the auditing server to listen on a port

   NOTE: We recommend you to use Sentinel as your auditing server.

3. Create a new security group with the following configuration and associate with the instance where the audit server is running:

Port	Source	Description
TCP (Auditing server port)	Private subnet	Allows receiving events from OpenText eDirectory servers

4. Update the following in `/etc/opt/novell/eDirectory/conf/auditlogconfig.properties` file on all the OpenText eDirectory instances:

   ```
   log4j.appender.S.Host=<Auditing server ip>
   log4j.appender.S.Port=<auditing server port>
   ```

5. Enable the corresponding CEF events from OpenText Identity Console. For more information, see Configuring the CEF Events for Auditing (https://www.netiq.com/documentation/edirectory-93/edir_admin/data/t44e7j6b8ufi.html#t44e7j6cx14i). Enabled events will be forwarded to the auditing server.

Disaster Recovery

Disaster recovery is performed in case of an instance crash where OpenText eDirectory was running. Perform the following steps for disaster recovery:

1. Stop the instance which has crashed and dissociate the EBS volume from it. For more information, see Detaching an Amazon EBS Volume from an Instance.
2. Configure a new EC2 instance with the same operating system as the instance which has crashed.
3. Install the same version of OpenText eDirectory in the new EC2 instance.
4. Attach the EBS volume to the new instance and mount the file system. For more information, see Attaching an Amazon EBS Volume to an Instance.
5. Bind mount the directories.

   To bind mount the directories, update the following in `/etc/fstab`:

```
<mount_point>/eDirectory_data /var/opt/novell/eDirectory none
defaults,bind 0 0

<mount_point>/nici_data /var/opt/novell/nici none defaults,bind 0 0

<mount_point>/eDirectory_nici_conf /etc/opt/novell/eDirectory none
defaults,bind 0 0
```

6 Change the IP address in `/etc/opt/novell/eDirectory/conf/nds.conf` to current instance IP address.

7 Upgrade OpenText eDirectory skipping health check. For more information, see *Upgrading OpenText eDirectory* .

8 Repair the network addresses using `ndsrepair` utility. For more information, see *DSRepair Options* in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.

9 Modify the CRL distribution point IP address if the tree CA IP address is changed. For more information on how to change the IP address, see Viewing and Modifying a CRL Configuration Object's Properties (https://www.netiq.com/documentation/edirectory-93/edir_admin/data/b1j4tpo3.html#b1j581ua) in the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.

10 Repair the server's default certificates using the OpenText Identity Console certificate server plug-in. To repair the default certificates, perform the following steps:

10a Login to OpenText Identity Console as Administrator.

10b Go to **Certificates Management** > **Default Certificate**.

10c Select the server(s) which owns the certificates and click **Next**.

10d Select **Yes All Default Certificates will be overwritten** and click **Next**.

10e Review the tasks to be performed and select **Finish** > click **OK**.

11 Configure LDAP and HTTP services to use new ECDSA Certificates.

Sizing Guidance for OpenText eDirectory Deployment on AWS

This guide recommends the best suited AWS EC2 Instance and EBS Volume for OpenText eDirectory deployment on AWS VPC. To help you estimate your sizing requirements, we have provided the most useful estimation factors along with various performance scenarios, which are tested under the controlled lab environment with specific configuration parameters in place.

- "Get Started with the AWS EC2 Instance Selection" on page 128
- "Details of Performance Test Environments" on page 128
- "Determining the Sizing Requirement" on page 130

Get Started with the AWS EC2 Instance Selection

Before deploying OpenText eDirectory on AWS, you must determine the EC2 Instance and EBS Volume types for achieving the optimal performance out of your OpenText eDirectory server. The performance of OpenText eDirectory depends on multiple resources regarding how an enterprise plans to use it. Some of the resources to take into consideration for your estimation are:

- **AWS EC2 Instance Category:** You must select the appropriate instance category required for your environment. We've considered the following Instance categories for the various performance scenarios in our lab environment:
  - General Purpose
  - Memory Optimized
  - Compute Optimized

  It is also recommended to use EBS-optimized EC2 instances for deploying OpenText eDirectory to avail a dedicated bandwidth between the EC2 instance and the attached EBS volume. It also enables an effective use of the IOPS which is provisioned on an EBS volume. For some instances, EBS optimization is not enabled by default. In such scenario, You must enable it manually.

- **AWS EC2 Instance Type:** After selecting the appropriate instance category for your environment, an instance type needs to be selected as per the business need or expected load. AWS provides a wide selection of EC2 Instance Types under each Instance Category. There are sub-categories under each category depending on various parameters, such as the type of processors or network throughput. The number of CPU cores, volume of RAM, network capacity and the cost depend on the instance type that is selected. For more information on Instance Types, see Amazon EC2 Instance Types.

- **AWS EBS Volume Types:** Since the root volume or the local storage in the EC2 instance is perishable, you must attach an AWS Elastic Block Storage (EBS) Volume to your EC2 instance to store OpenText eDirectory DIB and other server files that need to be persisted. EBS volumes allow you to persist the stored data even after the AWS EC2 instance is terminated, which helps in data recovery scenarios. Data stored in the EBS volume is replicated by default within the same Availability Zone. For more information, see EBS Volume Types.

  OpenText eDirectory being a read intensive directory service, the selection of the EBS Volume type will depend on high disk I/O. Due to this, it is recommended to use SSD volume types over HDD volume types because the SSD volume types handle small or random disk I/O more efficiently than the HDD volume types, which is basically used for streaming the disk I/O.

  Provisioned IOPS SSD volume type (io1 and io2) allow you to specify a consistent IOPS rate while creating the volume types when compared to General Purpose SSD volume type (gp2) where performance might be inconsistent. Also, the General Purpose SSD will tie up configuration of IOPS with the disk size which does not exist with the Provisioned IOPS SSD volume types.

Details of Performance Test Environments

In this section, we explain the various configuration parameters, instance categories and types that have been used in our controlled lab environment.

EC2 Instance Types:

The following instance types have been used during the performance tests:

Table 6-3 *Instance Type 4xLarge*

Category	Type	CPU	Memory	EBS Optimized	EBS Bandwidth
Compute optimized	c5a.4xlarge	16 CPU, 3.3 GHz	32 GB	Yes	3,170 Mbps

Table 6-4 *Instance Type 2xLarge*

Category	Type	CPU	Memory	EBS Optimized	EBS Bandwidth
General Purpose	m5a.2xlarge	8 CPU, 2.5 GHz	32 GB	Yes	2,880 Mbps
Memory optimized	r5a.2xlarge	8 CPU, 2.5 GHz	64 GB	Yes	2,880 Mbps
Compute optimized	c5a.2xlarge	8 CPU, 3.3 GHz	16 GB	Yes	3,170 Mbps

Table 6-5 *Instance Type xLarge*

Category	Type	CPU	Memory	EBS Optimized	EBS Bandwidth
General Purpose	m5a.xlarge	4 CPU, 2.5 GHz	16 GB	Yes	2,880 Mbps
Memory optimized	r5a.xlarge	4 CPU, 2.5 GHz	64 GB	Yes	2,880 Mbps
Compute optimized	c5a.xlarge	4 CPU, 3.3 GHz	8 GB	Yes	3,170 Mbps

Common Parameters

Table 6-6 *Explanation of Common Parameters*

Parameter	Description
AWS VPC Details	One Public Subnet and one Private Subnet. Both Client and OpenText eDirectory are in VPC Private subnet.

Parameter	Description
OpenText eDirectory EC2 Instance Details	◆ **Operating System**: SUSE Linux Enterprise Server 15 SP2 (HVM), SSD Volume Type ◆ **OpenText eDirectory build**: OpenText eDirectory 9.2 ◆ **Elastic Block Storage**: Provisioned IOPS SSD io2, 100 GB, IOPS – 1000 ◆ **Dib**: 3 GB, Over 2 million users
Operations	◆ **Operation**: LDAPS (636) Search and Modify operations ◆ **Percentage**: 80% Read 20% Write operations. ◆ **Search scope**: subtree search ◆ **Search base**: ou=users,o=novell ◆ **Search filter**: Random user under Search base ◆ **Attributes**: sn;title
OpenText eDirectory Tuning	**OpenText eDirectory Flaim DB setting:** preallocatecache=true cache=4000000000 cpinterval=360
LDAP Client EC2 Instance	◆ **Operating System**: RHEL8.0 ◆ **LDAP Client tool**: JMeter 5.1 ◆ **EC2 Instance type**: m5d.4xlarge, 16 CPU, 64 GB RAM ◆ **TCP Port range**: 1024-65534

Determining the Sizing Requirement

In this section, we present our recommendations based on performance data for commonly used scenarios. This data will help you to determine the optimal AWS EC2 Instance for your environment. Before you start with the appropriate sizing estimation for your OpenText eDirectory deployment, you must identify the best suited Instance Category for your deployment.

Scenario # 1 - Identifying the AWS EC2 Instance Category

In this performance scenario, we'll determine the appropriate AWS EC2 Instance Category for OpenText eDirectory deployment and operations.

The below graphs depict a measurement of Average Response Time (in milliseconds) and Operations per second respectively against the number of Threads. Measurement has been taken from three instance categories i.e., General Purpose, Memory Optimized and Compute Optimized that belong to 2xlarge Instance Type.

Figure 6-2 *Change in average response time based on number of threads performed concurrently for m5a vs r5a vs c5a*

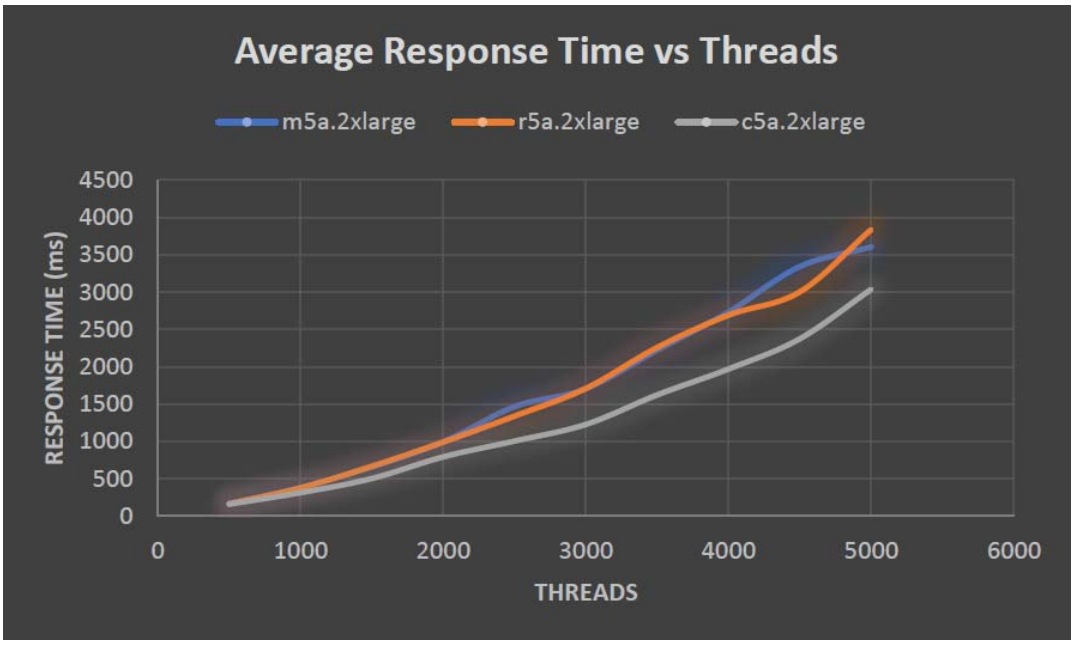
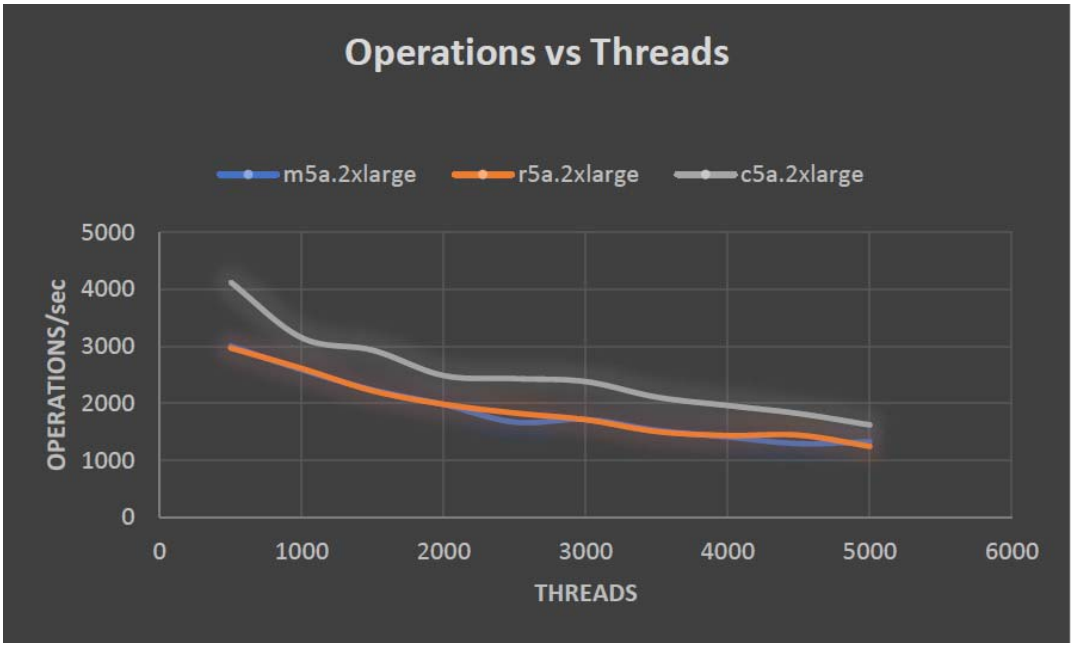


Figure 6-3 *Change in number of operations performed concurrently based on the number of threads*



The below graphs depict a measurement of Average Response time (in milliseconds) and Operations per second respectively against number of Threads. Measurement was taken from three instance categories i.e., General Purpose, Memory Optimized and Compute Optimized that belongs to xlarge type.

Figure 6-4 *Change in average response time based on number of threads performed concurrently for m5a vs r5a vs c5a*

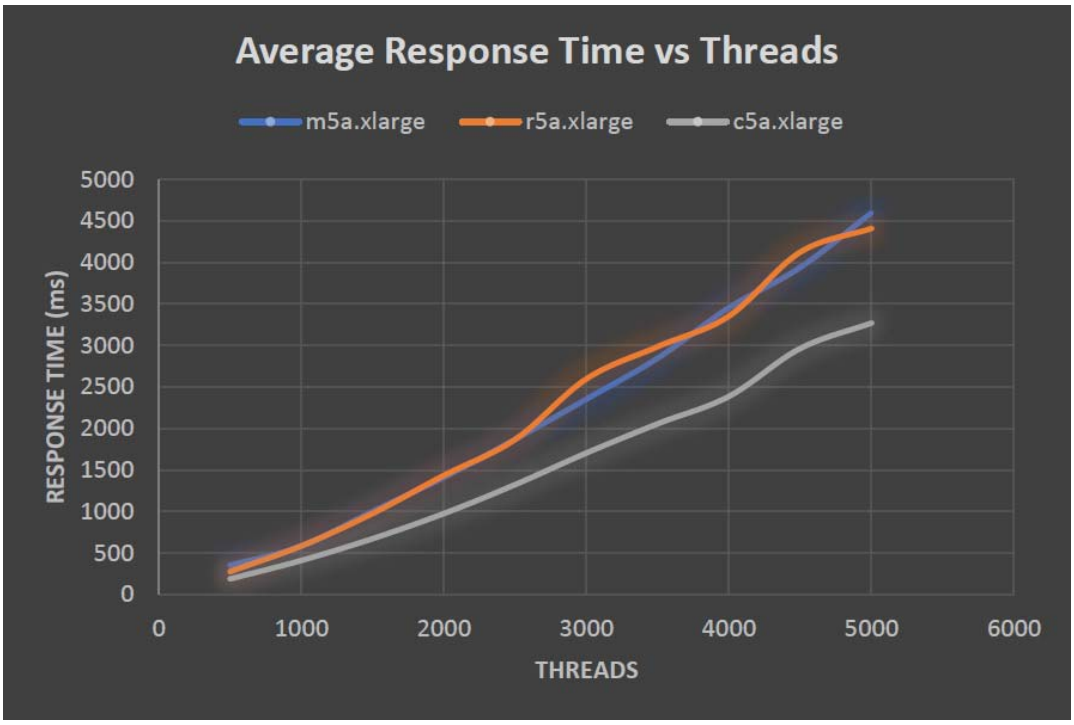
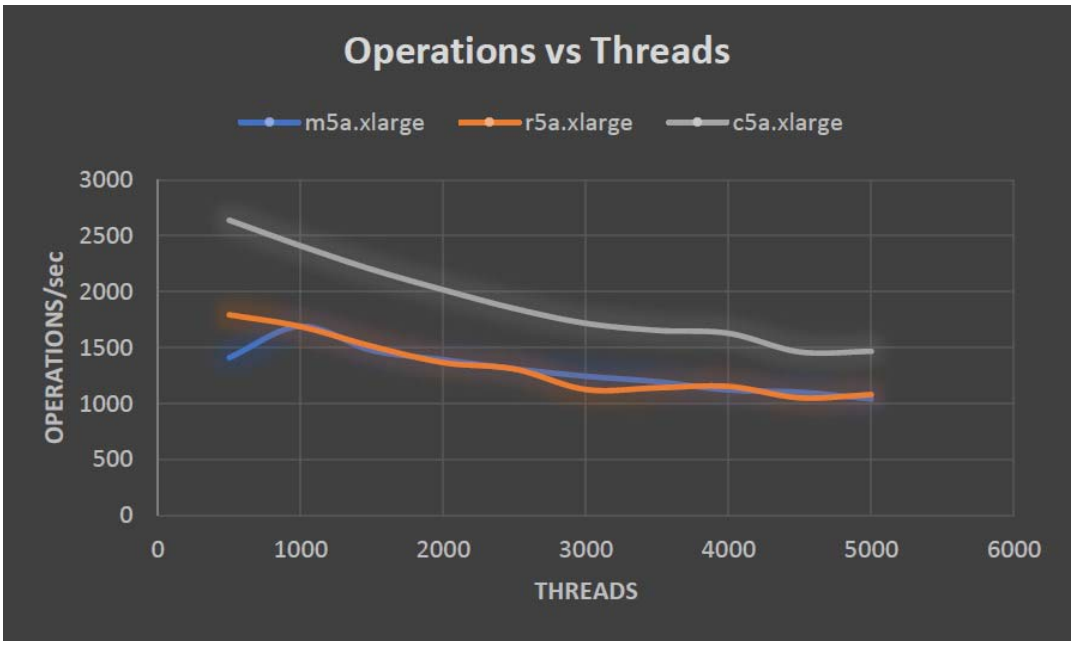


Figure 6-5 *Change in number of operations performed concurrently based on the number of threads*



Recommendation

As per the above test scenario, you must use the Compute Optimized instances along with EBS Volume to get the best OpenText eDirectory performance when deployed in an AWS environment. Instance type must be chosen initially based on the expected load. You can always resize the Instance type if it doesn't meet the resource requirements. You must also attach the Provisioned IOPS SSD volume to the EC2 instance for optimal IO and throughput performance.

Scenario # 2 - Identifying the AWS EC2 Sizing Requirement

In the previous performance scenario, we determined that Compute Optimized is the appropriate AWS EC2 Instance Category for OpenText eDirectory deployment and operations.

In this performance test scenario, the data has been collected from three different Compute Optimized instance types, i.e., xlarge, 2xlarge and 4xlarge to help you with the best suited sizing guidenace.

The below graphs depict a measurement of Average Response time (in milliseconds) and Operations per second respectively against the number of Threads. For this test, 500 milliseconds were considered as the maximum acceptable average response time.

Figure 6-6 *Change in average response time based on number of threads performed concurrently for xlarge vs 2xlarge vs 4xlarge*

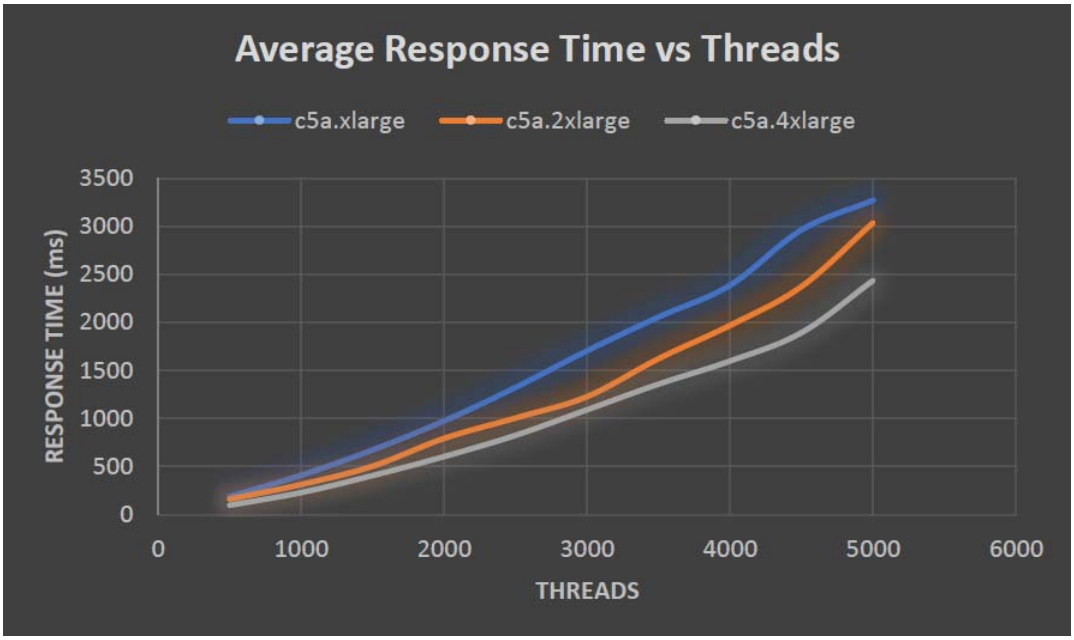
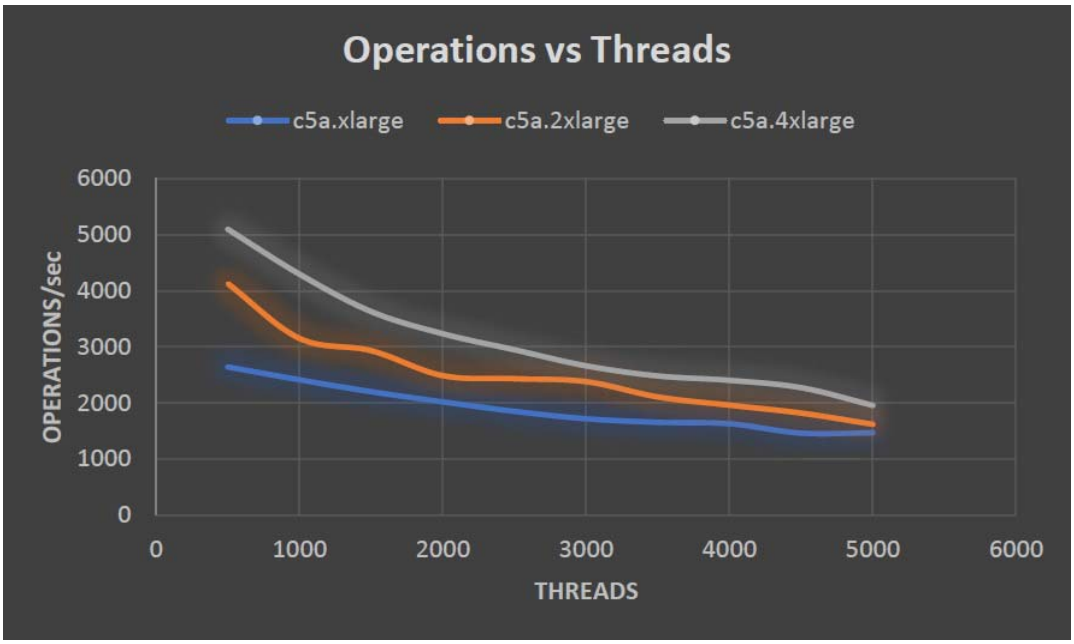


Figure 6-7 *Change in number of operations performed concurrently based on the number of threads*



Recommendation

The following table shows the measured maximum number of threads and the corresponding operations per second possible, to keep the average response time within the acceptable limit of 500 milliseconds for each of the specified EC2 instances.

Instance Type	Threads	Operations per Second
c5a.xlarge	1200	2300
c5a.2xlarge	1500	2900
c5a.4xlarge	1750	3500

You can select the best suited Compute instance size based on their expected Thread and throughput values as explained above.

7 Deploying OpenText eDirectory Using Docker Container

This chapter explains how to deploy OpenText eDirectory using Docker container.

- "Why Docker?" on page 135
- "Planning to Deploy OpenText eDirectory Using Docker Container" on page 135
- "Deploying OpenText eDirectory Container" on page 136
- "Post-Deployment Tasks" on page 142
- "Managing OpenText eDirectory Data Storage" on page 145
- "Upgrading OpenText eDirectory Using Docker Container" on page 145

Why Docker?

Docker is the most common application containerization technology. It is a platform designed to create, deploy, and run applications using containers. Containers encapsulate an application with its own operating system and all other dependencies, such as libraries and packages. Deploying OpenText eDirectory using Docker containers has the following advantages:

- **High Portability:** Any application running in containers can be deployed easily to any Docker supported operating systems and hardware platforms.
- **Easy to Deploy:** Containers allow applications to be more rapidly deployed, upgraded or even scaled through Orchestration tools.
- **Consistency:** There will be no impact on the functionality of OpenText eDirectory regardless of where the containers are deployed.

For more information on Docker and its components, see, Docker Overview.

Planning to Deploy OpenText eDirectory Using Docker Container

This section explains the system requirements and prerequisites for deploying OpenText eDirectory Docker container.

System Requirements

Platform Requirements

❒ Docker Community Edition version 18.06 and above is sufficient for deploying OpenText eDirectory Docker container.

- ❒ `overlay2` is the recommended Docker storage driver. BTRFS is not a supported file system of the host on which Docker can be installed.
- ❒ Linux kernel version 3.10 or higher.
- ❒ openSUSE Leap 15.4 works efficiently with the Docker Community Edition version 20.10.9-ce or similar.

Hardware Requirements

- ❒ A minimum of 4 GB RAM and 30 GB Hard disk space is to be provisioned on the Docker Host machine.

NOTE: Memory, CPU and Hard Disk requirements will vary depending on the type of deployment and the number of containers to be deployed. Always provision more resources than the current requirement to handle any possible scale up in future.

Prerequisites

- ❒ Docker host machines should be configured with a Static IP Address.
- ❒ Docker should be installed. For more information on supported platforms, see Docker Documentation.
- ❒ Docker daemon should be up and running.
- ❒ OpenText eDirectory Docker image tarball should be downloaded from the Software License and Download portal.
- ❒ Users that need to perform container administration on Docker, should be added to the `docker` group.

Docker CLI

The explanation of various commands used in Docker CLI is found here.

Deploying OpenText eDirectory Container

OS base image of the OpenText eDirectory Docker image is openSUSE Leap 15.4. OpenText eDirectory image archive file should be downloaded to the Docker Host machine. After downloading the archive file, it has to be extracted and then the image has to be loaded into the local Docker registry by using the following commands:

```
tar -xvf eDirectory_93x_Container.tar.gz
docker load --input eDirectory_93x/eDirectory_93x.tar.gz
```

The above command will load a Docker image named `edirectory:9.3.x`.

OpenText eDirectory Docker container accepts all parameters of the `ndsconfig` utility with the Docker Run command. For information on `ndsconfig` utility, see, “Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server” on page 49.

NOTE: Setting the password using `-w` option of the `ndsconfig` utility in the Docker Run command is not recommended. Password set using this option can be viewed in plain-text using Docker inspect command. Specifying the admin FDN and admin password in the prompt is a more secure way of configuring the admin credential.

OpenText eDirectory Docker container uses default values for the following `ndsconfig` parameters. Hence, you should not configure these parameters in the Docker run command:

- **Configuration File:** /config/eDirectory/inst/conf/nds.conf
- **Instance Location:** /config/eDirectory/inst/data/data
- **DIB Location:** /config/eDirectory/inst/data/data/dib

NOTE: It is important that the instance data and configuration is populated under the `/config` folder of the container. This is to enable persistent storage and upgrade functionality of OpenText eDirectory containers. For more information, see "Managing OpenText eDirectory Data Storage" on page 145.

The default log file location of the OpenText eDirectory container is `/config/eDirectory/inst/data/log`.

Before deploying OpenText eDirectory, you must consider the following recommendations:

- Docker containers do not have any resource constraints by default. This provides every container with the access to all the CPU and memory resources provided by the host's kernel. You must also ensure that one running container should not consume more resources and starve other running containers by setting limits to the amount of resources that can be used by a container.
  - Docker container should ensure that a Hard Limit is applied for the memory used by the container using the `--memory` flag on Docker run command.
  - Docker container should ensure that a limit is applied to the amount of CPU used by a running container using the --`cpuset-cpus` flag on the Docker run command.
- `--pids-limit` should be set to 300 to restrict the number of kernel threads spawned inside the container at any given time. This is to prevent DoS attacks.
- You must set the container restart policy to `on-failure` with number of retries as 5 using the `--restart` flag on Docker run command. Containers will have to be manually restarted if the Docker daemon on the host machine gets restarted.
- You must only use the OpenText eDirectory container once the health status shows as **healthy** after the container comes up. To check the container's health status, run the following command:

  ```
  docker ps --filter status="running"
  ```

- Docker containers usually have a default list of Linux capabilities enabled. You must ensure to keep only the following capabilities enabled for OpenText eDirectory container and drop the others:
  - AUDIT_WRITE
  - CHOWN
  - DAC_OVERRIDE

- SETGID
- SETUID
- NET_BIND_SERVICE
- SYS_CHROOT (Only if enabling SLP service)
- SYS_PTRACE (Only if using utilities that make use of Linux `ptrace`. Such as `gdb`)

For more information on how to add and drop capabilities, see Runtime privilege and Linux capabilities (https://docs.docker.com/engine/containers/run/#runtime-privilege-and-linux-capabilities).

- eDirectory container will always start as non-root user (`nds`). As an additional security measure, enable user namespace remapping on the daemon to prevent privilege-escalation attacks from within the container. For more information on user namespace remapping, see Isolate containers with a user namespace.
- You must set the `--stop-timeout` option in the Docker run command with a value of 180. This is to make sure that the nds service gets enough time to shutdown gracefully.

NOTE: If you are using any previous version of standalone OpenText eDirectory, you will not be able to migrate your setup to the Docker environment using OpenText eDirectory 9.3 Docker container.

OpenText eDirectory Docker container supports Host and Overlay network drivers for deployment in a multi-host Docker environment:

- "Deploying OpenText eDirectory Container in Host Network" on page 138
- "Deploying OpenText eDirectory Container in User Defined Overlay Network" on page 140

Deploying OpenText eDirectory Container in Host Network

OpenText eDirectory containers can be deployed in a Hybrid environment using the Host network driver only on Linux. For information on Docker networks, see Configure Networking.

NOTE: Host Network is not supported on Windows.

A Hybrid environment is a combination of both legacy and container based deployments of OpenText eDirectory servers in the same tree. A Hybrid network enables a seamless introduction of OpenText eDirectory Docker containers to an existing production environment that is already hosting a legacy OpenText eDirectory deployment. In Docker Host networking, service ports cannot be re-used as the network stack of the host is shared by both legacy and containerized eDirectory deployments. Also, a containerized OpenText eDirectory server will appear as a legacy OpenText eDirectory server to clients and other servers in the tree.

The following example shows, how to create a new tree using OpenText eDirectory container:

```
docker run -it --name eDir-container-1 --stop-timeout 180 --restart on-failure:5 --memory="700M" --cpuset-cpus="1" --pids-limit="300" --volume eDir-volume1:/config --network=host edirectory:9.3.0 new -t docker-tree1 -n novell -S m1 -B 164.99.1.1@524 -o 1028 -O 1030 -L 389 -l 636 --configure-eba-now yes
```

The following example shows, how to add an OpenText eDirectory container replica server to an existing tree:

```
docker run -it --name eDir-container-2 --stop-timeout 180 --restart on-
failure:5 --memory="700M" --cpuset-cpus="1" --pids-limit="300" --volume
eDir-volume2:/config --network=host edirectory:9.3.0 add -t docker-tree1 -
n novell -S m2 -B 164.99.10.10@2524 -o 2028 -O 2030 -L 2389 -l 2636 --
configure-eba-now yes -p 164.99.1.1@1524
```

For more information on EBA, see Viewing Information About EBA in OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.

The below list describes the parameters to add an OpenText eDirectory container replica server to an existing tree.

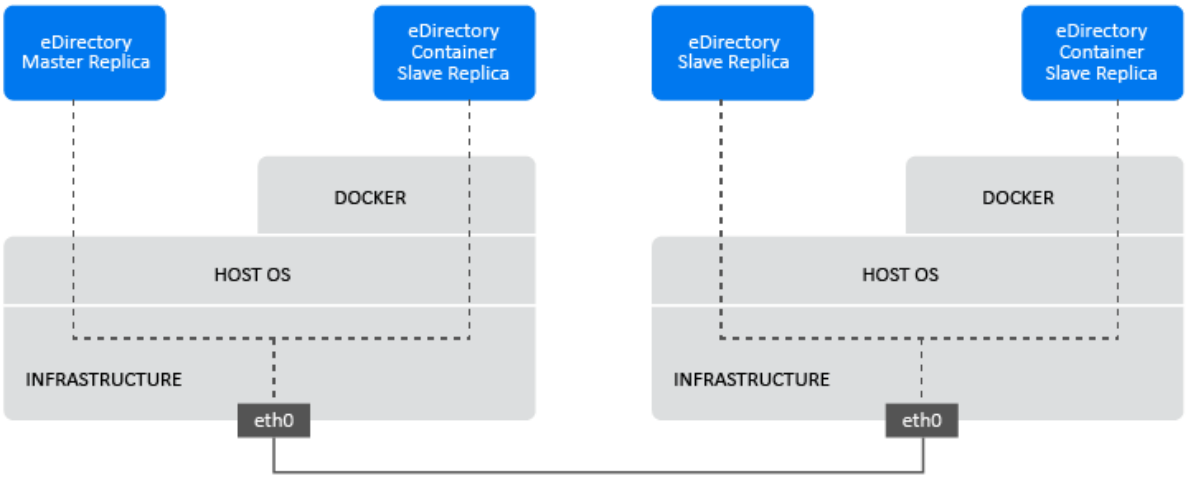
Table 7-1 *Description of the configuration parameters*

Configuration Parameters	Description
--name eDir-container-1	Name of the docker container
--volume eDir-volume1	Name of the docker volume used to store OpenText eDirectory data
edirectory:9.3.0	Docker image you downloaded and loaded. Note the version number – no warning that this is probably not the actual version you downloaded
-t docker-tree1	OpenText eDirectory tree name
-n novel	OpenText eDirectory container where the server object will be placed
-S m1	Name of server object that will be created in OpenText eDirectory
-B 164.99.1.1	IP address of docker host
-o 1028	OpenText eDirectory HTTP port
-O 1030	OpenText eDirectory HTTPS port
-L 389OpenText	eDirectory LDAP port
-l 636	OpenText eDirectory LDAPS port
--configure-eba-now Yes	Yes – install EBA now

NOTE:

- `--network` flag is used to deploy the container using the Host networking driver.
- Port numbers of services should not be repeated across OpenText eDirectory containers running on the same Docker host.
- IP addresses used in the above commands are of the Docker host machine where the container is supposed to run.

Figure 7-1 *Deploying OpenText eDirectory Container in Host Network*



Deploying OpenText eDirectory Container in User Defined Overlay Network

A User Defined Overlay network can be used to create a distributed network of OpenText eDirectory containers running on multiple Docker daemon hosts. OpenText eDirectory container in user defined overlay network can be deployed in both Linux and Windows. Docker Swarm service should be used to join the Docker hosts to a Swarm, so that OpenText eDirectory containers running on them can communicate seamlessly. For more information on Docker Overlay network driver, see Use Overlay Networks.

NOTE: Scaling and scheduling functions of a Docker swarm is not certified with OpenText eDirectory containers. Migration of OpenText eDirectory containers across hosts by Swarm service is not also supported.

Prerequisites

- ❒ A Docker swarm should be created with at least one Docker host configured as `manager` and the other hosts as `workers`.
- ❒ Create an attachable Overlay network called `myOverlay`.
- ❒ Open the following ports on the firewall between Docker hosts for cluster management and communication within a Docker Swarm:
  - TCP port 2377
  - TCP and UDP port 7946
  - UDP port 4789
- ❒ Containers deployed in Overlay network should be assigned a static internal IP address that belongs to the `myOverlay` subnet.

For information on how to deploy a Swarm and create a user defined Overlay network, see Networking with overlay networks.

Before deploying OpenText eDirectory container in a user defined Overlay network, you must consider the following recommendations:

- OpenText eDirectory container master replica server and it's R/W replicas must be deployed within the same Overlay network. Communication with other standalone OpenText eDirectory servers or containers running outside the Overlay network will not be supported.
- It is recommended to deploy the Identity Console Docker container within the same user defined Overlay network for administering OpenText eDirectory. For more information on how to deploy Identity Console Docker containers, see Installing OpenText Identity Console as Docker Container.
- You can find the network details of the user defined overlay network by running the following command:

  ```
  docker inspect myOverlay
  ```

The following command shows how to create a new tree using OpenText eDirectory container:

```
docker run -it --name eDir-container-1 --stop-timeout 180 --restart on-
failure:5 --memory="700M" --cpuset-cpus="1" --pids-limit="300" --volume
eDir-volume1:/config --network=myOverlay --ip=10.0.0.5 edirectory:9.3.0
new -t docker-tree1 -n novell -S m1 –B @524 -o 8028 -O 8030 -L 389 -l 636 -
-configure-eba-now yes
```

The following command shows how to obtain the IP Address of the OpenText eDirectory container created above:

```
docker inspect eDir-container-1 --format
{{.NetworkSettings.Networks.myOverlay.IPAddress}}
```

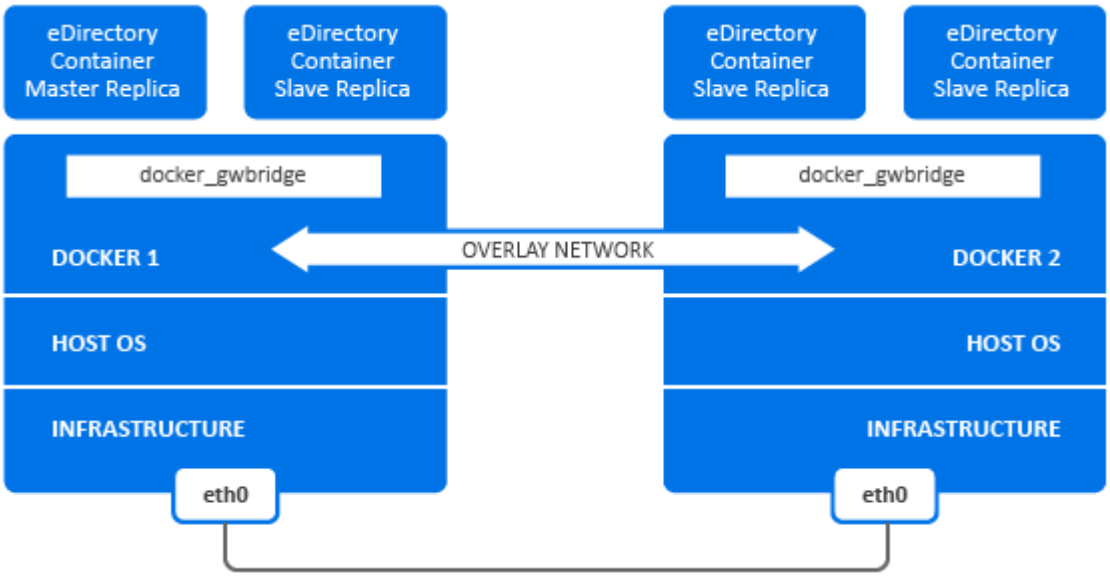
The displayed IP Address can be used as the `remote_IP_Address` while adding an OpenText eDirectory container replica server to the Tree.

The following command shows how to add an eDirectory container replica server to an existing tree:

```
docker run -it --name eDir-container-2 --stop-timeout 180 --restart on-
failure:5 --memory="700M" --cpuset-cpus="1" --pids-limit="300" --volume
eDir-volume2:/config --network=myOverlay --ip=10.0.0.6 edirectory:9.3.0
add -t docker-tree1 -n novell -S m2 –B @524 -o 8028 -O 8030 -L 389 -l 636 -
-configure-eba-now yes -p <remote_IP_Address>
```

NOTE:
- `--network` flag is used to deploy the container in the user-defined Overlay network called `myOverlay` using the Overlay network driver.
- In the above examples, `--ip` flag is used to assign a static internal IP address to the container that belongs to `myOverlay` subnet.
- `-B @524` option will allow the NCP, http and https services to bind to all the available interfaces.

Figure 7-2 *Deploying OpenText eDirectory Container in User Defined Overlay Network*



Post-Deployment Tasks

The following tasks should be performed after deploying OpenText eDirectory containers.

- “Executing Commands on a Running OpenText eDirectory Container” on page 142
- “Configuring OpenSLP for OpenText eDirectory Docker Container” on page 143
- “Installing NMAS Methods in OpenText eDirectory Docker Container” on page 143
- “Installing New Packages in OpenText eDirectory Docker Container” on page 144
- “Configuring the CEF Property File in OpenText eDirectory Docker Container” on page 144

NOTE: The OS base image of the OpenText eDirectory Docker image is openSUSE Leap with SysVinit as its init system. You must add all the environment variables required for the OpenText eDirectory service to run in the `pre_ndsd_start` file.

Executing Commands on a Running OpenText eDirectory Container

Run the following command in the Docker host machine to get a bash shell in the OpenText eDirectory Docker container:

```
docker exec -it eDir-container-1 /bin/bash
```

The above command will set the OpenText eDirectory binary path to `/opt/novell/eDirectory/bin`.

NDS utility commands can be executed on the container prompt. An example is shown below:

```
nds@abbae7c93b1c:~> ndsstat
```

```
Tree Name: EDR93E
Server Name: .CN=Master.O=server.T=EDR93E.
Binary Version: 40301.00
Root Most Entry Depth: 0
Product Version: OpenText eDirectory for Linux x86_64 v9.3 [DS]
```

The above command can be executed directly from the host machine. An example is shown below:

```
docker exec -it eDir-container-1 /opt/novell/eDirectory/bin/ndsstat
```

```
Tree Name: EDR93E
Server Name: .CN=Master.O=server.T=EDR93E.
Binary Version: 40301.00
Root Most Entry Depth: 0
Product Version: OpenText eDirectory for Linux x86_64 v9.3 [DS]
```

Configuring OpenSLP for OpenText eDirectory Docker Container

Perform the following steps to start the SLP server in a running OpenText eDirectory container:

1. Start `slpd` by running the following command:

   ```
   docker exec --user root eDir-container-1 /usr/sbin/slpd
   ```

2. Restart OpenText eDirectory by running the following commands:

   ```
   docker exec eDir-container-1 /opt/novell/eDirectory/bin/ndsmanage
   stopall
   ```

   ```
   docker exec eDir-container-1 /opt/novell/eDirectory/bin/ndsmanage
   startall
   ```

NOTE: ◆Stopping and Restarting the container will stop the SLP daemon. It has to be restarted manually. If a stale PID file is present at `/var/run/slpd.pid`, the PID file should be removed before starting the daemon.

- A stale PID file is a file having PID of a stopped or killed process (in our case SLP daemon process).
- In an Overlay environment, the SLP DA should be running within the same Overlay network.
- If you do not configure OpenSLP, you might see the error message `-626`.

Installing NMAS Methods in OpenText eDirectory Docker Container

Perform the following tasks to install NMAS methods in OpenText eDirectory containers:

NOTE: The NMAS methods are available at `/home/nds/eDirectory/nmas` by default.

---

1. Login to the OpenText eDirectory container by running the following command:

   ```
   docker exec -it eDir-container-1 bash
   ```

2. Add NMAS method:

```
cd /home/nds/eDirectory/nmas/NmasMethods/Novell/<method-name>

nmasinst -addmethod admin.novell docker-tree1 ./config.txt
```

NOTE: For more information on adding NMAS methods, see Using the nmasinst Utility to Install a Login Method in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

3 Exit from the container console:

```
exit
```

4 Restart the OpenText eDirectory container:

```
docker restart eDir-container-1
```

Installing New Packages in OpenText eDirectory Docker Container

The base OS included in the OpenText eDirectory image contains all dependencies for the normal functioning of the OpenText eDirectory application in a container environment. In case a new utility or a debugging tool has to be introduced which is not included by default in the image, then it has to be separately brought in to the running container.

Run the following command to install a new package in a running OpenText eDirectory container using `zypper`:

```
docker exec --user root -it eDir-container-1 zypper in <package>
```

Alternatively, new packages can be installed after copying them directly to the container from the Docker host machine using the `docker cp` command. For more information, see, docker cp.

Configuring the CEF Property File in OpenText eDirectory Docker Container

You must perform the following changes in the CEF property file to enable Cache and Rolling file logging. The CEF property file (`auditlogconfig.properties`) is located in `/config/eDirectory/inst/conf` folder.

```
# Cache location directory
# Directory should be available for creating cache files
log4j.appender.S.CacheDir=/config/eDirectory/inst/data

# Log file for appender R.
log4j.appender.R.File=/config/eDirectory/inst/data/log/cef-events.log
```

For more information, see Configuring the CEF Property File in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

NOTE: To configure Event System Caching, set the environment variables as shown below in the `pre_ndsd_start` file located in `/opt/novell/eDirectory/sbin` folder of the OpenText eDirectory container:

```
export NDSD_EVENT_DISK_CACHE=1
export NDSD_EVENT_DISK_CACHE_DIR=/config/eDirectory/inst/data/data
```

Managing OpenText eDirectory Data Storage

Docker Volume is the preferred mechanism for persistently storing OpenText eDirectory data and configuration. For more information on persistent storage, see Manage data in Docker.

OpenText eDirectory application data that requires persistent storage will be placed under the `/config` directory in the container during startup. A Docker volume has to be mounted to the `/config` path in the OpenText eDirectory container to persistently store the data on the Docker host file system outside the container. Even if a container is stopped or removed for administrative purposes, application data inside the volume is retained.

This practice is useful for retaining old configuration and data during an upgrade of OpenText eDirectory container. For information on upgrading eDirectory container, see "Upgrading OpenText eDirectory Using Docker Container" on page 145.

The following example shows how to create a Docker volume called `eDir-volume-1`:

```
docker volume create eDir-volume-1
```

The following example shows how to start an OpenText eDirectory container with a volume mounted to it for storage purpose:

```
docker run -it --name eDir1-Host --restart on-failure:5 --memory="700M" --
cpuset-cpus="1" --pids-limit="300" --volume eDir-volume1:/config --
network=host edir930:latest new -t docker-tree1 -n novell -S m1 -B
164.99.179.213@1524 -o 1028 -O 1030 -L 1389 -l 1636 --configure-eba-now yes
```

In the above command, `eDir-volume1` is the Docker volume that is created and mounted to `/config` location in the OpenText eDirectory container.

Upgrading OpenText eDirectory Using Docker Container

When a new version of OpenText eDirectory Image is available, the administrator can perform an upgrade procedure to deploy container with the latest version of OpenText eDirectory. Ensure to store all necessary application related data persistently in Docker volumes before performing an upgrade. Perform the following steps to upgrade OpenText eDirectory using Docker container:

1. Stop and remove the running eDirectory container. Since the running containers cannot use the new image, they should be stopped and removed before performing an upgrade.
2. Start a new container using the new OpenText eDirectory Docker image and the application data of the old container stored in Docker volume:

   The following example shows how to start an OpenText eDirectory container with a volume mounted to it for storage purpose:

   ```
   docker run -it --name eDir1-Host --stop-timeout 180 --restart on-
   failure:5 --memory="700M" --cpuset-cpus="1" --pids-limit="300" --volume
   eDir-volume1:/config --network=host <eDirectory_image> new -t docker-
   tree1 -n novell -S m1 -B <Host_IP_Address>@1524 -o 1028 -O 1030 -L 1389
   -l 1636 --configure-eba-now yes
   ```

The following example shows how to upgrade the OpenText eDirectory container created in step 2:

```
docker run -it --name eDir1-Host --stop-timeout 180 --restart on-
failure:5 --memory="700M" --cpuset-cpus="1" --pids-limit="300" --volume
eDir-volume1:/config --network=host <Latest_eDirectory_image> upgrade
```

`eDir-volume1` is the same volume that retains the application data of the old OpenText eDirectory container.

NOTE: ◆OpenText eDirectory container should only be upgraded after removing the container running the old version of the image.

- It is not recommended to use the `-a` and `-w` options of the `ndsconfig` command. You should use the on-screen prompt to enter the administrator's credentials for upgrading the container.

Recovering OpenText eDirectory Docker Containers

When a running OpenText eDirectory container becomes inaccessible, deleted or unusable due to unknown reasons, a container recovery should be performed. In such scenario, the impacted container should be stopped and removed. A new container should be started using the same OpenText eDirectory image and the Docker Volume of the impacted container. Perform the following steps to recover a OpenText eDirectory container:

1. Stop and remove the impacted container.
2. Start a new container with the same OpenText eDirectory image and the volume of the impacted container. The following example shows how to recover an impacted container:

   ```
   docker run -it --name eDir1-Host --stop-timeout 180 --restart on-
   failure:5 --memory="700M" --cpuset-cpus="1" --pids-limit="150" --volume
   eDir-volume1:/config --network=host <same_eDirectory_image> upgrade
   ```

   `eDir-volume1` is the same volume that retains the application data of the impacted OpenText eDirectory container.

8 Installing OpenText eDirectory on Linux and Windows with IPv6 Addresses

OpenText eDirectory 932 supports both IPv4 and IPv6 addresses. You can enable IPv6 addresses during the OpenText eDirectory installation process. If you are upgrading from a previous version, you must manually enable IPv6 addresses.

OpenText eDirectory 9.3 supports Dual IP stack, Tunneling, and Pure IPv6 transition methods. It supports only the global IP addresses. For example,

- [2015::12]
- [2015::12]:524

There is no change in the OpenText eDirectory functionality on IPv6 addresses from IPv4 addresses except that you must specify IPv6 addresses within square braces [ ]. Also, you can use hostname instead of an IP address. If you are using the hostname, you must specify it in the `etc\hosts` file and associate it with the IPv6 address.

The following are examples of some of the OpenText eDirectory utilities with IPv6 addresses:

```
ndsstat -h [6015:abc:def:123:456:12:0:123]
```

```
ndsstat -h [6015:abc:def:123:456:12:0:123]:524
```

```
ndslogin -h [2015::4] admin.organization
```

```
ndscheck -h [6015:abc:def:123:456:12:0:123] -a admin.organization -w
password
```

```
ldapadd -h [2015::4] -p 389 -D cn=admin,o=organization -w password -f
adduser.ldif
```

```
ldapdelete -h [6015:abc:def:123:456:12:0:123] -p 389 -D
cn=admin,o=organization -w password cn=user21,o=organization
```

```
ldapmodify -h [2015::4] -p 389 -D cn=admin,o=organization -w password -f
modify.ldif
```

```
ldapsearch -h [6015:abc:def:123:456:12:0:123] -p 389 -D
cn=admin,o=organization -w password -b o=organization
objectclass=inetorgperson
```

```
http://[2015::3]:8028/nds
```

OpenText eDirectory 9.3 does not support Link local, IPv4-mapped IPv6, and IPv4-compatible IPv6 address types.

The following sections describe how to install and configure OpenText eDirectory 9.3 on Linux and Windows where IPv6 is already configured:

- “Configuring OpenText eDirectory on Linux with IPv6” on page 148
- “Installing or Upgrading OpenText eDirectory on Windows with IPv6” on page 149

For information about the differences in the Linux and Windows platforms for IPv6, see "Troubleshooting IPV6 Issues".

Configuring OpenText eDirectory on Linux with IPv6

This section provides information about configuring OpenText eDirectory on a Linux computer that already supports IPv6 addresses:

Creating a New OpenText eDirectory Tree

You can configure a new OpenText eDirectory tree with an IPv6 address by passing the IPv6 address along with the `-B` option in the `ndsconfig` command. For example:

```
ndsconfig new -t CORP-TREE -B [2015::3]@524 -P ldap://[2015::3]:389,ldaps://[2015::3]:636
```

For the LDAP listeners to automatically start listening on the IPv6 addresses, you must specify the LDAP URLs with the `-P` option while configuring OpenText eDirectory. If you do not specify them during the initial configuration, you can add them in the `ldapInterfaces` attribute using the `ldapconfig` command or Identity Console after the initial configuration. For more information, see "Adding LDAP URLS for IPV6 on the LDAP Server Object" on page 149.

Adding a Server to an Existing OpenText eDirectory Tree

You can add a server to an existing tree with IPv6 by passing the IPv6 address with the `-B` option in the `ndsconfig` command. For example:

```
ndsconfig add -t CORP-TREE -B [2015::4]@524 -P ldap://[2015::4]:389,ldaps://[2015::4]:636
```

For the LDAP listeners to automatically start listening on the IPv6 addresses, you must specify the LDAP URLs with the `-P` option while configuring OpenText eDirectory. If you do not specify them during the initial configuration, you can add them in the `ldapInterfaces` attribute using the `ldapconfig` command or Identity Console after the initial configuration. For more information, see "Adding LDAP URLS for IPV6 on the LDAP Server Object" on page 149.

Enabling IPv6 Addresses on Existing or Upgraded OpenText eDirectory Servers

1 Add the IPv6 interface address with the port number in the `/etc/opt/novell/eDirectory/ conf/nds.conf` file. You must add it in each configuration file, if the computer has multiple instances configured.

The following are some examples:

```
n4u.server.interfaces=164.99.90.148@524,[2015::4]@524,[2015:1234:2345:3456:abcd:bcde:cdef:aaaa]@524
```

```
http.server.interfaces=164.99.90.148@8028,[2015::4]@8028,[2015:1234:2345:3456:abcd:bcde:cdef:aaaa]@8028
```

```
https.server.interfaces=164.99.90.148@8030,[2015::4]@8030,[2015:1234:2
345:3456:abcd:bcde:cdef:aaaa]@8030
```

2 Restart ndsd using the following commands:

```
ndsmanage stopall
ndsmanage startall
```

Adding LDAP URLS for IPV6 on the LDAP Server Object

If you do not specify the LDAP URLs during the initial OpenText eDirectory configuration, you can use the `ldapconfig` command or Identity Console to add them in the `ldapInterfaces` attribute.

The following are examples for using the `ldapconfig set` and the `ldapconfig -s` command:

```
ldapconfig set "ldapInterfaces=ldap://[2015::3]:389,ldaps://[2015::3]:636"
```

```
ldapconfig -s
"ldapInterfaces=ldap://[2015::3]:389,ldapInterfaces=ldaps://[2015::3]:636"
```

To add LDAP URLs in Identity Console:

1 Login to OpenText Identity Console, click **LDAP Configuration**.

2 On the **Search Results** menu > Select the LDAP server - <server name/CN>.

3 Click **Modify LDAP Server** > **Information** > LDAP interface .

4 On the **Add Item window** > add LDAP URLS in the LDAP Interfaces field.

5 Click **Add**, then click **Save**.

LDAP Server modified successfully! message appears, Click **OK**.

Installing or Upgrading OpenText eDirectory on Windows with IPv6

This section provides information about configuring OpenText eDirectory on a Windows computer that already supports IPv6 addresses.

Enabling IPv6 While Installing or Upgrading OpenText eDirectory

If you want to use IPv6 addresses, ensure that you select the **Enable IPv6** check box under **IPv6 Preference** during the OpenText eDirectory installation. If you select it, the DHost starts listening on the IPv6 addresses. If you do not enable IPv6 addresses during the installation process, and then decide to use them later, you must run the setup program again.

Enabling IPv6 for Existing Servers

If you want to use IPv6 addresses for an already configured OpenText eDirectory server, you must rerun the installation and select the **Enable IPv6** check box under **IPv6 Preference**. It enables the NCP, HTTP, and HTTPS protocols for the IPv6 addresses.

Accessing OpenText iMonitor

You can access OpenText iMontior over IPv6 addresses using the following link:

`http://[2015::3]:8028/nds`

9 Operating OpenText eDirectory in FIPS Mode

OpenText eDirectory 9.3 leverages the Federal Information Processing Standards (FIPS) compliant features to meet the security requirements of U.S. Federal agencies and customers with highly secure environments. This chapter provides information about configuring and operating OpenText eDirectory in the FIPS mode.

You can run OpenText eDirectory in FIPS 140-2 mode supported by NICI and OpenSSL modules.

- "Configuring OpenText eDirectory in FIPS Mode" on page 151

Configuring OpenText eDirectory in FIPS Mode

From OpenText eDirectory 9.3 onwards, by default FIPS mode is disabled on both NICI and OpenSSL in all the platforms. OpenText eDirectory provides switches to configure the FIPS mode to suit your requirement. When FIPS mode is enabled OpenText eDirectory server, all applications and modules running inside OpenText eDirectory including the OpenSSL will be in FIPS mode. For example, LDAP, HTTP, and all cryptographic operations in EBA. Operating OpenText eDirectory in FIPS mode does not allow communication over SSLv3 and restricts the cipher usage to high strength ciphers. For more information, see Configuring LDAP Objects and Configuring HTTP Server Object in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

You have to set only one flag to enable the FIPS mode for the OpenText eDirectory. This enables both NICI and OpenSSL.

Execute the following pre-requisites before enabling the NICI FIPS to make NDS login work:

1. Create - PBKDF2 (Password-Based Key Derivation Function 2) password policy. For more information see Understanding Non-Reversible Password Storage. Then, assign PBKDF2 password policy to all the users.
2. Set SCRAM as default login sequence. For more information see Password Authentication.
3. Once PBKDF2 policy is assigned, it is recommended that all the users must change their passwords, before enabling FIPS in NICI.
4. Enable NICI FIPS once the password of all the users are changed. To enable NICI in FIPS mode, see Using NICI for Configuring System-Level FIPS Mode .
5. TLS and FIPS in NICI are enabled based on the `nici.cfg` configuration.

To enable the FIPS mode in OpenText eDirectory for both NICI and OpenSSL:

- **Windows:** FIPS can be enabled on windows, setting the NICI Restriction Level flag to 1 in the windows registry. For more information, see Using NICI for Configuring System-Level FIPS Mode (https://www.netiq.com/documentation/edirectory-93/nici_admin_guide/data/agbdek3.html#b1habb0n).

- **Linux:** FIPS in NICI are enabled based on the `nici.cfg` configuration. For more information, see Using NICI for Configuring System-Level FIPS Mode (https://www.netiq.com/documentation/edirectory-93/nici_admin_guide/data/agbdek3.html#b1habb0n).

NOTE: It is not recommended to disable FIPS, once it is enabled.

---

10 Relocating the DIB

After installing and configuring OpenText eDirectory, if there is a need to relocate the DIB, you can do it. You might want to relocate your DIB for multiple reasons, such as, if the number of objects in the tree is expected to grow but the current file system where the DIB exists does not have sufficient space.

Linux

Complete the following procedure to relocate your DIB:

1. Check the server status by entering the following command at the command line:

   `ndscheck`

2. Stop the OpenText eDirectory service using ndsmanage as follows:

   2a Enter `ndsmanage` at the command prompt.

   2b Select the instance you want to stop.

   The menu expands to include the options you can perform on a specific instance.

   2c Enter k to stop the instance.

3. Get the current DIB location using the following command:

   `ndsconfig get n4u.nds.dir`

4. Copy the DIB to the new location as follows:

   `cp -rp current_location new_location`

   For example, to copy the DIB to `/home/nds/`, enter the following:

   `cp -rp /var/opt/novell/eDirectory/data//* /home/nds//`

5. Edit the instance-specific `nds.conf` configuration file and change the parameter value of `n4u.nds.dir` as follows:

   `n4u.nds.dir=new_location`

   For example, if you are changing the DIB from `/var/nds/` to `/home/nds/`, type the following:

   `n4u.nds.dir=/home/nds/`

6. Start the OpenText eDirectory service as follows

   6a Enter `ndsmanage` at the command prompt.

   6b Select the instance you want to start.

   The menu expands to include the options you can perform on a specific instance.

   6c Enter s to start the instance.

7 Check the server status as follows:

```
ndscheck
```

Windows

DIB relocation is currently not supported. However, you can locate the DIB in a custom location during the OpenText eDirectory installation.

11 Upgrade Requirements of OpenText eDirectory 9.3

One of the unique features of OpenText eDirectory is its ability to maintain the tight referential integrity. Any object Classes derived from Top will have a reference attribute in its class definition. This is a hidden attribute added to all the referenced objects that are internally maintained by OpenText eDirectory. Background processes keep running to check the links between the referenced object and the referencing objects.

If the referenced object is from a different partition than the one held locally in the server, an external reference to that object will be created locally in the external reference partition. An external reference is a representation of an object existing in the OpenText eDirectory tree. However, it is not a copy of the object and its assigned attributes.

Though we can remove the Reference attribute from OpenText eDirectory, currently, the class definitions are untouched to maintain the backward compatibility in the tree.

This chapter explains the changes and possible upgrade scenarios in OpenText eDirectory 9.3.

- "Reference Changes in 8.8 SP1 or Later Versions" on page 155
- "Upgrade Process in 9.3" on page 156

Reference Changes in 8.8 SP1 or Later Versions

The reference attribute is a hidden attribute and is maintained on each referenced object. This is created and maintained by DS. The new referencing code in DS is based on a Flexible Adaptable Information Manager (FLAIM) index called `LocalEntryIDIndex` that DS creates. Though FLAIM maintains the index, the usage is determined by DS. FLAIM automatically updates the index when a DN value is added or deleted. Each key in the index is a compound key, i.e, DN of the object being referenced + Entry ID of the referencing object. For example, if there is an object with Entry ID 343, and it has a "member" value that points to object #899, FLAIM will automatically generate a key in the index of 899+343. DS can now do lookups in the index to find all the objects pointing to object #899. Object #899 does not have to keep a reference attribute on itself to remember all the objects referencing it. Actually, FLAIM maintains the index without knowing how it is used, but DS has the code that knows how to use the index.

NOTE: It is recommended not to directly upgrade OpenText eDirectory from version 8.8SP1 to 9.3.You must upgrade to 9.2 and then to 9.3. Upgrading OpenText eDirectory from any previous version of 8.8 SP1 to 9.2 impacts the performance of your OpenText eDirectory server. This occurs because all the indexes will be recreated automatically. If you are upgrading from 8.8 SP1 and later versions, this issue will not occur.

However, the new way of maintaining references requires a database upgrade when the existing OpenText eDirectory instance is upgraded to 8.8 SP1 or later versions. The upgrade requires the creation of a new index, which will require traversing each entry in the database. It also requires the

removal of all of the “reference” attributes from each entry in the database. In addition, some internal octet string attributes used by DS that had embedded DNs would need to generate some new DN values to store alongside the octet string value. All this would be a time consuming process on a large database. Since DS is changed to do referential integrity using new FLAIM feature, and that depends on the new index, there is no way DS can really operate until the conversion is complete. Therefore, the first time an existing database is opened, all reference attributes need to be changed to a new index. It could take hours before it actually opens and is ready for use by applications for a large database.

Upgrade Process in 9.3

The `ndsconfig upgrade` command is used to upgrade the necessary configuration of the components such as HTTP, LDAP, SNMP, SAS, and NMAS.

Using the Force Option to Upgrade OpenText eDirectory From Older Versions on Linux

OpenText eDirectory 9.3 only supports upgrades from version 9.2 or later on Linux.

To upgrade from OpenText eDirectory versions from 8.8.8, upgrade to OpenText eDirectory 9.2 and then upgrade to eDirectory 9.3. After upgrading to eDirectory 9.2, run the repair certificate in the identity console and then upgrade to OpenText eDirectory 9.3. For more information on repairing the default certificate, see (https://www.netiq.com/documentation/identity-console/identity_console-admin/data/t4eksjvptege.html)

IMPORTANT: If the Identity Manager changelog module is already installed, then you must set the environment variable `NDSD_IGNORE_IDM_CHECK` to 1 while upgrading OpenText eDirectory to 9.2. For example:

- On Linux: `NDSD_IGNORE_IDM_CHECK=1 ./nds-install`
- On Windows: Set `NDSD_IGNORE_IDM_CHECK` to `true` before running `setup.exe`.

---

12 Configuring OpenText eDirectory on Linux

OpenText eDirectory includes configuration utilities that simplify the configuration of various OpenText eDirectory components on Linux computers. The following sections provide information about functionality and usage of OpenText eDirectory configuration components:

- "Configuration Utilities" on page 157
- "Configuration Parameters" on page 159
- "Security Considerations" on page 165

Configuration Utilities

This section provides information about using the following OpenText eDirectory configuration utilities:

- "The ndsconfig Utility" on page 157
- "Using LDAP Tools to Configure the LDAP Server and LDAP Group Objects" on page 158
- "Using the nmasinst Utility to Configure NetIQ Modular Authentication Service" on page 158
- "Customizing OpenText eDirectory" on page 158

The ndsconfig Utility

You can use the ndsconfig utility to configure OpenText eDirectory. This utility can also be used to add the OpenText eDirectory Replica Server into an existing tree or to create a new tree. For more information, see "Using the ndsconfig Utility to Add or Remove the OpenTexte Directory Replica Server" on page 49.

NOTE: ◆Ensure that the NCP server name is unique in the network.

To change the current configuration of the installed components, use the following syntax:

```
ndsconfig {set value_list | get [parameter_list] | get help
[parameter_list]}
```

Refer to "Configuration Parameters" on page 159 for a description of `ndsconfig` parameters.

IMPORTANT: After installation, ensure that you run the ndsconfig utility from the installed location on the server, which is `/opt/novell/eDirectory/bin` by default. Do not run ndsconfig from the installation package.

Using LDAP Tools to Configure the LDAP Server and LDAP Group Objects

You can use the LDAP tools included with OpenText eDirectory on Linux computers to modify, view, and refresh the attributes of LDAP Server and Group objects.

For more information, see "Using LDAP Tools on Linux" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

Using the nmasinst Utility to Configure NetIQ Modular Authentication Service

For OpenText eDirectory 9.3, by default, ndsconfig configures NMAS. You can also use nmasinst to configure NMAS.

ndsconfig only configures NMAS and does not install the login methods. To install these login methods, you can use nmasinst. For more information, see "Using the nmasinst Utility to Configure NMAS" on page 63.

Customizing OpenText eDirectory

- "Using the ndsd init Script" on page 158
- "Using OpenText eDirectory on SLES 15 and RHEL 8 Platforms" on page 159

Using the ndsd init Script

The ndsd init script starts the daemon when the system starts with the configuration parameters from the default configuration file, `/etc/opt/novell/eDirectory/conf/nds.conf`.

NOTE: You should not use the `/etc/init.d/ndsd` script in the `systemd` environment . `Systemd` is currently supported with SLES 15 and RHEL 8 and RHEL 9 and above platforms only. For more information, see "Using OpenText eDirectory on SLES 15 and RHEL 8 Platforms" on page 159.

Before starting ndsd, ensure that any SLP (Service Location Protocol) agent is running on the host. You can install OpenSLP, any native SLP available with your operating system, or NetIQ SLP.

NOTE: To start OpenText eDirectory, use the ndsmanage utility.

NOTE: Run the following commands to start and stop OpenText eDirectory in SLES 15 and RHEL8 or later.

- To start ndsd, run `systemctl start ndsd*`
- To stop ndsd, run `systemctl stop ndsd*`

The OpenText eDirectory configuration creates the following shell scripts in the `/opt/novell/eDirectory/sbin` location:

- `pre_ndsd_start`

- post_ndsd_start
- pre_ndsd_stop
- post_ndsd_stop

As the name indicates, the pre_ndsd_start script is executed before the ndsd binary is started by the /etc/init.d/ndsd script. The post_ndsd_start script is executed after the ndsd binary is started by the /etc/init.d/ndsd script. Similarly, the pre_ndsd_stop and post_ndsd_stop scripts are executed before and after stopping the ndsd process, respectively.

You can add commands of your choice to these scripts to get them executed. By default, the post_ndsd_start script has commands to ensure that /etc/init.d/ndsd comes out after ensuring that the LDAP services are up and running.

NOTE: You must add all the environment variables required for the OpenText eDirectory service in the env_custom script which is located in the /etc/opt/novell/eDirectory/conf directory. Exporting of environment variables on terminals or /etc/init.d/ndsd script is not used by OpenText eDirectory. For more information about the environment variables, see TID 7018431.

Using OpenText eDirectory on SLES 15 and RHEL 8 Platforms

OpenText eDirectory starts the daemon when the system starts with the configuration parameters from the default configuration file, /etc/opt/novell/eDirectory/conf/nds.conf.

Before starting ndsd, ensure that any SLP (Service Location Protocol) agent is running on the host. You can install OpenSLP, any native SLP available with your operating system, or OpenText SLP.

To start or stop OpenText eDirectory, use the ndsmanage utility.

The OpenText eDirectory configuration creates the following shell scripts in the /opt/novell/eDirectory/sbin location:

- pre_ndsd_start_custom: Use this script for custom addition of commands before executing eDirectory.
- post_ndsd_start_custom: Use this script for custom addition of commands after executing eDirectory.
- post_ndsd_stop_custom: Use this script for custom addition of commands after stopping eDirectory.

NOTE:
- Do not use any of the factory scripts from the /opt/novell/eDirectory/sbin / location. The OpenText eDirectory configuration will use the factory scripts. To include additional commands of your choice, use custom scripts.
- After upgrading the operating system, run the ndsconfig upgrade utility.

Configuration Parameters

The OpenText eDirectory configuration parameters are stored in the nds.conf file.

When configuration parameters are changed, ndsd needs to be restarted for the new value to take effect. You should use ndsmanage to restart ndsd.

However, for some configuration parameters, ndsd need not be restarted. These parameters are listed below:

- `n4u.nds.inactivity-synchronization-interval`
- `n4u.nds.synchronization-restrictions`
- `n4u.nds.janitor-interval`
- `n4u.nds.backlink-interval`
- `n4u.nds.drl-interval`
- `n4u.nds.flatcleaning-interval`
- `n4u.nds.server-state-up-thresholdn4u.nds.heartbeat-scheman4u.nds.heartbeat-data`

The following table provides a description of all the configuration parameters.

Parameter	Description
`n4u.nds.preferred-server`	The host name of the machine that hosts the OpenText eDirectory service. Default = null
`n4u.base.tree-name`	The tree name that Account Management uses. This is a mandatory parameter set by the Account Management Installer. This parameter cannot be set.
`n4u.base.dclient.use-udp`	DClient can use UDP in addition to TCP for communicating with the OpenText eDirectory servers. This parameter enables the UDP transport feature. Default = 0 Range = 0, 1
`n4u.base.slp.max-wait`	The Service Location Protocol (SLP) API calls timeout. Default = 30 Range = 3 to 100 This value is in seconds. This option is supported only by OpenText SLP and not OpenSLP.
`n4u.nds.advertise-life-time`	OpenText eDirectory reregisters itself with the Directory Agent after this time period. Default = 3600 Range = 1 to 65535 This value is in seconds.

Parameter	Description
n4u.server.signature-level	Determines the level of enhanced security support. Increasing this value increases security, but decreases performance. Default = 1 Range = 0 to 3
n4u.nds.dir	The OpenText eDirectory directory information database. Default: /var/opt/novell/eDirectory/data/ This parameter cannot be set using the ndsconfig set command. You can manually change this parameter if you want to relocate your DIB. However, we do not recommend you do so.
n4u.nds.server-guid	A globally unique identifier for the OpenText eDirectory server. Default = null
n4u.nds.server-name	The name of the OpenText eDirectory Server. Default = null
n4u.nds.bindery-context	The Bindery context string. Default = null
n4u.nds.server-context	The context that the OpenText eDirectory server is added to. This parameter cannot be set or changed.
n4u.nds.external-reference-life-span	The number of hours unused external references are allowed to exist before being removed. Default = 192 Range = 1 to 384
n4u.nds.inactivity-synchronization-interval	The interval (in minutes) after which full synchronization of the replicas is performed, following a period of no change to the information held in the OpenText eDirectory on the server. Default = 60 Range = 2 to 1440
n4u.nds.synchronization-restrictions	The Off value allows synchronization with any version of the OpenText eDirectory. The On value restricts synchronization to version numbers you specify as parameters. For example, ON,420,421. Default = Off

Parameter	Description
n4u.nds.janitor-interval	The interval (in minutes) after which the OpenText eDirectory Janitor process is executed. Default = 2 Range = 1 to 10080
n4u.nds.backlink-interval	The interval (in minutes) after which the OpenText eDirectory backlink consistency is checked. Default = 780 Range = 2 to 10080
n4u.nds.drl-interval	The interval (in minutes) after which the OpenText eDirectory distributed reference link consistency is checked. Default = 780 Range = 2 to 10080
n4u.nds.flatcleaning-interval	The interval (in minutes) after which the flatcleaner process automatically begins purging and deleting entries from the database. Default = 720 Range = 1 to 720
n4u.nds.server-state-up-threshold	The server state up threshold, in minutes. This is the time after which the OpenText eDirectory checks the server state before returning -625 errors. Default = 30 Range = 1 to 720
n4u.nds.heartbeat-schema	The heartbeat base schema synchronization interval in minutes. Default = 240 Range = 2 to 1440
n4u.nds.heartbeat-data	The heartbeat synchronization interval in minutes. Default = 60 Range = 2 to 1440
n4u.nds.dofsync	Setting this parameter to 0 increases update performance significantly for large databases, but there is a risk of database corruption if the system crashes.
n4u.server.configdir	The OpenText eDirectory configuration files are placed here. Default = /etc

Parameter	Description
`n4u.server.vardir`	The OpenText eDirectory and utilities log files are placed here. Default = `/var/opt/novell/eDirectory/log`
`n4u.server.libdir`	The eDirectory specific libraries are placed here in the `nds-modules` directory. Default = `/opt/novell/eDirectory/lib`
`n4u.server.sid-caching`	Enables SSL session ID caching. Refer to the SSL v3.0 RFC for more details about session ID caching in SSL.
`n4u.server.tcp-port`	The default port used if the port number is not specified in the `n4u.server.interfaces` parameter.
`n4u.server.interfaces`	The IP address and port number that OpenText eDirectory server should listen on for client connections. The value can be a comma-separated list specifying more than one combination of possible settings. For example: `n4u.server.interfaces=101.1.2.3@524,100.1.2.3@1524`
`n4u.server.max-interfaces`	This parameter specifies maximum number of interfaces that OpenText eDirectory will use. Default = 128 Range = 1 to 2048
`n4u.server.max-openfiles`	This parameter specifies the maximum number of file descriptors that OpenText eDirectory can use. Default = maximum allowed by the administrator
`n4u.server.max-threads`	The maximum number of threads that will be started by the OpenText eDirectory server. This is the number of concurrent operations that can be done within the OpenText eDirectory server. Default = 64 Range = 32 to 512 Refer to the *Troubleshooting* to set an optimum value.
`n4u.server.idle-threads`	The maximum number of idle threads that are allowed in the OpenText eDirectory server. Default = 8 Range = 1 to 128
`n4u.server.start-threads`	Initial number of threads to be started up. Default = 8

Parameter	Description
n4u.server.log-levels	This parameter helps to configure the error logging settings for the server-side messages. It sets the message log level to LogFatal, LogWarn, LogErr, LogInfo, or LogDbg.
n4u.server.log-file	This parameter specifies the log file location where the messages would be logged. By default, the messages are logged into the ndsd.log file.
n4u.ldap.lburp.transize	Number of records that are sent from the OpenText Import/Export client to the LDAP server in a single LBURP packet. You can increase the transaction size to ensure that multiple add operations can be performed in a single request. Default = 25 Range = 1 to 250
n4u.server.listen-on-loopback	It is a boolean parameter, and enabled by default. In a few recent Linux distributions, the hostname in the /etc/hosts file is associated with the loopback address. Though the common address given in the SLES systems is 127.0.0.2, it can be anything from 127.0.0.0 to 127.255.255.255 (valid loopback addresses).
http.server.interfaces	Comma-separated list of interfaces that HTTP server should use.
http.server.request-io-buffer-size	Default IO buffer size.
http.server.request_timeout-seconds	Server request timeout.
http.server.keep-timeout-seconds	Number of seconds to wait for the next request from the same client on the same connection.
http.server.threads-per-processor	HTTP thread pool size per processor.
http.server.session-exp-seconds	Session expiration time in seconds.
http.server.sadmin-passwd	Session administrator password.
http.server.module-base	HTTP server webroot.
https.server.cached-cert-dn	HTTPS server cached certificate DN.
https.server.cached-server-dn	HTTPS server cached DN.
http.server.trace-level	Diagnostic trace level of HTTP server.
http.server.auth-req-tls	HTTP server authentication requires TLS.
http.server.clear-port	Server port for the HTTP protocol.

Parameter	Description
`http.server.tls-port`	Server port for the HTTPS protocol.
`n4u.server.fips`	Specifies whether OpenText eDirectory server runs in FIPS mode. Default = 1. This means OpenText eDirectory runs in FIPS mode. To disable the FIPS mode, pass n4u.server.fips=0 with ndsconfig set command and restart the server.

NOTE: For more details information on the OpenText eDirectory configuration parameters, refer to the `nds.conf` man page.

Security Considerations

The following security considerations are recommended:

- Make sure that only authenticated users have browse rights to the tree. To limit this, do the following:
  - Remove browse rights of [Public] on tree root.
  - Assign [Root] browse rights on tree root.
- Set the `ldapBindRestrictions` attribute on the LDAP server object to `Disallow anonymous Simple Bind`. This prevents the clients from doing anonymous binds.

13 Deploying OpenText eDirectory on High Availability Clusters

The primary method through which OpenText eDirectory supports high availability is by configuring multiple servers through synchronization. However, clustering may be a more viable alternative for achieving high availability in some environments.

This section provides guidelines for configuring OpenText eDirectory on high availability clusters by using shared storage. The information in this section is generalized for shared storage high availability clusters on supported Windows and Linux platforms, and the information is not specific to a particular cluster manager.

State data for OpenText eDirectory must be located on the shared storage so that it is available to the cluster node that is currently running the services. This means that the OpenText eDirectory DIB must be located on the cluster shared storage. The root OpenText eDirectory instance on each of the cluster nodes must be configured to use the DIB on the shared storage.

In addition to the DIB, it is also necessary to share NICI data so that server-specific keys are replicated among the cluster nodes. NICI data used by all cluster nodes must be located on the cluster shared storage.

Other OpenText eDirectory configuration and log data should also reside on shared storage.

OpenText eDirectory 9.3 includes a utility for both Linux and Windows servers that automatically configures OpenText eDirectory in your clustered environment, including copying data to a specified shared storage location, updating the appropriate configuration parameters, and setting up OpenText eDirectory services on cluster nodes other than the primary node.

The procedures in the following sections are based on the following assumptions:

- You are familiar with OpenText eDirectory installation procedures.
- You are using a two-node cluster.

NOTE: A two-node cluster is the minimum configuration used for high availability. However, the concepts in this section can easily be extended to a cluster with additional nodes. Note that OpenText eDirectory does not support load balancing by using multiple cluster nodes.

This section covers the following topics:

- “Clustering OpenText eDirectory Services on Linux” on page 168
- “Clustering OpenText eDirectory Services on Windows” on page 171
- “Troubleshooting Clustered Environments” on page 173
- “Configuration Utility Options” on page 174

Clustering OpenText eDirectory Services on Linux

This section describes how to configure OpenText eDirectory 9.3 by using high availability clustering on Linux.

- "Prerequisites" on page 168
- "Installing and Configuring OpenText eDirectory" on page 168
- "Configuring SNMP Server in Clustered Linux Environments" on page 170

Prerequisites

- Two or more Linux servers with clustering software
- External shared storage supported by the cluster software, with sufficient disk space to store all OpenText eDirectory and NICI data
- Virtual IP address
- OpenText eDirectory 9.1 or later

NOTE: The nds-cluster-config utility only supports configuring the root OpenText eDirectory instance. OpenText eDirectory does not support configuring multiple instances and non-root installations of OpenText eDirectory in a cluster environment.

Installing and Configuring OpenText eDirectory

1 Install and configure OpenText eDirectory on the server you want to use as the primary cluster node. For more information on installation and configuration procedures, refer to the "Using the nds-install Utility to Install OpenText eDirectory Components" on page 41.

NOTE:
- When configuring OpenText eDirectory, the default NCP server name is the host server name of the computer on which you installed OpenText eDirectory. Because OpenText eDirectory is hosted on multiple hosts in a clustered environment, however, you should specify an NCP server name that is unique to the cluster instead of using the default name. For example, you can specify the name `clusterserver` for the NCP server when you configure OpenText eDirectory on the primary cluster node.
- During the configuration process, ensure you set the virtual IP address for your OpenText eDirectory installation. In a clustered environment, OpenText eDirectory only listens on the virtual IP address, not on the system IP address.

---

2 After you install and configure eDirectory, navigate to the `nds.conf` file, which is located in the `/etc/opt/novell/eDirectory/conf`.

3 Edit the `nds.conf` file to set the value of the `n4u.nds.preferred-server` setting to the virtual IP address of the clustered installation, then save and close the file.

4 Verify the OpenText eDirectory installation by using the `ndsstat` command.

OpenText eDirectory must be up and running on the primary cluster node.

5 Mount the shared file system by using the cluster manager.

6 Back up all data in the following directories before running the configuration utility:

- `/var/opt/novell/nici`
- `/var/opt/novell/eDirectory/data` (`n4u.server.vardir`)
- `/var/opt/novell/eDirectory/data/dib` (`n4u.nds.dibdir`)
- `/etc/opt/novell/eDirectory/conf` (`n4u.server.configdir`)
- `/var/opt/novell/eDirectory/log`

NOTE: If you install OpenText eDirectory in a non-default location, you can use the `ndsconfig get` command to find the `vardir`, `dir` paths used in your installation. `nds.conf` should be in the default location, which is `/etc/opt/novell/eDirectory/conf/nds.conf`.

---

7 On the primary cluster node server, open a terminal and run the following command to stop the OpenText eDirectory service:

```
ndsmanage stopall
```

8 In the terminal, navigate to the location of the configuration utility, `nds-cluster-config`. The utility is located in the `/opt/novell/eDirectory/bin` directory.

9 Run the following command:

```
nds-cluster-config -s /<sharedfilesystem>
```

Where <*sharedfilesystem*> is the location you want to use for the eDirectory shared cluster data.

NOTE: You can also run the utility in unattended mode by using the `-u` option. If you use this option, the utility does not ask for confirmation when configuring OpenText eDirectory on a cluster.

If you use the unattended option, you must also use the `-s` option and specify the shared cluster file system.

---

10 After the utility verifies the cluster shared storage is valid, click **y** to continue with configuration on the cluster.

The configuration utility moves the data in the directories above to the following locations on the shared file system:

- <*sharedfilesystem*>`/nici`
- <*sharedfilesystem*>`/data`
- <*sharedfilesystem*>`/data/`
- <*sharedfilesystem*>`/conf`
- <*sharedfilesystem*>`/log`

11 Start OpenText eDirectory services by running the following command:

```
ndsmanage startall
```

12 Check the status of OpenText eDirectory by using ndsstat. OpenText eDirectory services should be up and running.

13 Stop OpenText eDirectory services by running the following command:

```
ndsmanage stopall
```

14 Log in to the server you want to use as the secondary node of the cluster.

15 Use the cluster manager to move the shared storage to the secondary node.

16 Install the same version of OpenText eDirectory on the secondary cluster node that you installed on the primary cluster node, but do not configure OpenText eDirectory.

17 In the terminal, navigate to the location of the configuration utility on the secondary node. The utility is located in the `/opt/novell/eDirectory/bin` directory.

18 Open a terminal and run the following command:

```
nds-cluster-config -s /<sharedfilesystem>
```

Where <*sharedfilesystem*> is the cluster shared storage. The path of the <*sharedfilesystem*> should be same as the path location specified when the primary node was configured.

The nds-cluster-config utility links the secondary cluster node to the shared OpenText eDirectory data located on the shared cluster file system.

19 Start OpenText eDirectory services by running the following command:

```
ndsmanage startall
```

Verify the status of eDirectory by using the `ndsstat` command.

20 Stop OpenText eDirectory services on the secondary node by running the `ndsmanage stopall` command.

21 After successfully configuring OpenText eDirectory on both nodes of the cluster, you must also change the startup mode of the ndsd service on each node by using the following command:

```
chkconfig -d ndsd
```

22 After the configuration utility finishes configuring the secondary node, you can use the cluster manager to add the OpenText eDirectory services in the cluster.

For more information on the cluster services on Linux, refer to the following documentation:

- SUSE Linux Enterprise Server (SLES 12 & above)
- SLES 11 SP4

IMPORTANT: Ideally, the cluster manager checks that the same DIB is not accessed by two or more nodes simultaneously. However, you must ensure that ndsd does not run from two or more cluster nodes simultaneously. This is because accessing the same DIB through two or more nodes leads to DIB corruption.

Configuring SNMP Server in Clustered Linux Environments

1 On all the nodes, modify the `snmpd.conf` file. For more information, see "Installing and Configuring SNMP Services for OpenText eDirectory" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

2 Start ndssnmpsa.

3 Select Yes as the `Remember password` option.

4 To start the snmp service, perform either of the following:

- Add `/etc/init.d/ndssnmpsa start` to the `post_ndsd_start` script and `/etc/init.d/ndssnmpsa stop` to the `pre_ndsd_stop` script.
- Add `ndssnmpsa` as a cluster resource with a dependency on eDirectory resource.

NOTE: Because OpenText eDirectory is listening on a virtual IP address, traps have the IP address of the host, which is the Agent IP address.

Clustering OpenText eDirectory Services on Windows

This section describes how to configure OpenText eDirectory 9.3 by using high availability clustering on Windows.

- "Prerequisites" on page 171
- "Installing and Configuring OpenText eDirectory" on page 171
- "Configuring SNMP Server in Clustered Windows Environments" on page 173

Prerequisites

- Two or more Windows servers with clustering software
- External shared storage supported by the cluster software
- Virtual IP address
- OpenText eDirectory 9.3

Installing and Configuring OpenText eDirectory

1 Install and configure OpenText eDirectory on the server you want to use as the primary cluster node. For more information on installation and configuration procedures, refer to the "Installing or Updating OpenText eDirectory 9.3 on a Windows Server" on page 72.

2 Mount the shared volume by using the cluster manager.

3 Back up all DIB files and NICI data before running the configuration utility.

4 On the primary cluster node, open a terminal and navigate to the `NDSCons.exe` utility. The utility is located in the <*`eDirectory installation folder`*> folder by default.

5 In the terminal, run the following command:

```
NDSCons.exe
```

6 In the NDSCons utility, click **Shutdown** to stop all eDirectory services.

7 Click **Yes** to confirm.

8 In the terminal, navigate to the location of the configuration utility, `dsclusterconfig.exe`. The utility is located in the <OpenText eDirectory installation folder> folder by default.

9 Run the following command:

```
dsclusterconfig.exe -s /<sharedfilesystem>
```

Where <*sharedfilesystem*> is the location you want to use for the OpenText eDirectory shared cluster data.

NOTE: ◆You can also run the utility in unattended mode by using -s with `-u` option.

- You must specify a folder within a shared drive mounted on the primary cluster node. You cannot specify only a drive name. For example, instead of specifying `E:`, you must specify `E:\Novell`.

---

10 After the utility verifies the cluster shared storage is valid, click **y** to continue with configuration on the cluster.

The configuration utility moves the data in the directories above to the following locations on the shared file system:

- <*sharedfilesystem*>/nici
- <*sharedfilesystem*>/Files

In addition to moving eDirectory data to the shared file system, the utility copies the OpenText eDirectory service registry key to the shared volume, saving the key as the file `ndsConfigKey`.

The utility also changes the Startup Type of the `NDS Server` service on the primary node computer from `Automatic` to `Manual`.

11 In the NDSCons utility, click **Startup** to start all eDirectory services.

12 Verify that all OpenText eDirectory services are running, then use the NDSCons utility to stop services again.

13 Close the NDSCons utility.

14 Log in to the server you want to use as the secondary node of the cluster.

15 Use the cluster manager to move the shared storage to the secondary node.

16 Use the OpenText eDirectory installer to perform an unattended installation of OpenText eDirectory on the secondary node. Ensure that the mode of installation is `install`.

17 In the terminal, navigate to the location of the configuration utility on the secondary node. The utility is located in the OpenText eDirectory installation folder by default.

18 Run the following command:

```
dsclusterconfig.exe -s /<sharedfilesystem>
```

Where <*sharedfilesystem*> is the cluster shared storage. The path of the <*sharedfilesystem*> should be same as the path location specified when the primary node was configured.

19 The dsclusterconfig utility updates registry on the secondary cluster node to the shared OpenText eDirectory data located on the shared cluster file system.

20 After the configuration utility finishes configuring the secondary node, open the NDSCons utility.

21 In the NDSCons utility, click **Startup**.

22 Click **Yes** to confirm.

23 When NDSCons starts all OpenText eDirectory services, verify OpenText eDirectory, then click **Shutdown**.

24 Click **Yes** to confirm.

25 To configure OpenText eDirectory in the Cluster Resource group, create a new resource in the Resource Group to be used for OpenText eDirectory.

You must provide the following details:

- Resource type - Generic Service
- Dependent on - IP address and shared disk in the Resource Group
- Service name - `NDS Server0`
- No start parameters
- Registry keys - `SYSTEM\CurrentControlSet\Services\NDS Server0`

NOTE: Ideally, the cluster manager checks that the same DIB is not accessed by two or more nodes simultaneously. However, you must ensure that ndsd does not run from two or more cluster nodes simultaneously. This is because accessing the same DIB through two or more nodes leads to DIB corruption.

Configuring SNMP Server in Clustered Windows Environments

1 On the primary cluster node, configure the master agent and set the startup type to automatic. For more information, see "Installing and Configuring SNMP Services for OpenText eDirectory" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

2 Save the OpenText eDirectory password when it prompts for the password.

3 Start the sub-agent.

4 Perform Step 1 to Step 3 on the other nodes.

Troubleshooting Clustered Environments

Repairing or Upgrading OpenText eDirectory on Clustered Nodes

While you perform a repair or upgrade on any of the cluster nodes, the other cluster nodes must be paused or on standby to ensure that automatic fail over does not occur.

Creating Windows Registry Keys

As part of the configuration process in clustered Windows environments, the configuration utility automatically creates a registry key, `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NDS Server0\ImagePath`, on the cluster shared file system. eDirectory needs the registry key in order to start the `x86 NDS Server` service on the cluster nodes.

If the utility cannot create the registry key and returns an error message during configuration, you must use the Registry Editor to manually create the registry key on all cluster nodes, even if the configuration utility appears to have successfully completed the configuration.

Create the following registry key on all nodes:

`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NDS Server0\ImagePath`

Assign the following value to the `ImagePath` key:

```
"<primarynodeinstallfolder>\NDS\ndsserv.exe" /
DataDir="<sharedstorage>\Files" ds
```

Where <*primarynodeinstallfolder*> is the folder where you installed eDirectory on the primary node and <*sharedstorage*> is the path to the shared file system location.

Configuration Utility Options

The options available for use in the configuration utility are as follows:

<*configuration utility*> [-h] [-u] [-s /<*sharedfilesystem*>]

Where <*configuration utility*> is either `nds-cluster-config` or `dsclusterconfig.exe`, depending on the platform, and <*sharedfilesystem*> is the location you want to use for the OpenText eDirectory shared cluster data.

Parameter	Description
`-h`	Displays the configuration utility help.
`-s`	Specifies the shared directory path for the cluster.
`-u`	Enables the utility to configure OpenText eDirectory on the cluster in unattended mode. If you run the utility by using the `-u` option, you must also use the `-s` option and specify the shared directory path. For example: `nds-cluster-config -u -s <sharedfilesystem>`

14 Uninstalling OpenText eDirectory

This chapter contains the following information:

- "Uninstalling OpenText OpenText eDirectory on Windows" on page 175
- "Uninstalling OpenText eDirectory on Linux" on page 180
- "Unattended Uninstallation of OpenText eDirectory on Linux" on page 181
- "Caveats for Uninstalling OpenText eDirectory" on page 181

Uninstalling OpenText OpenText eDirectory on Windows

Use the Windows Control Panel to remove OpenText eDirectory, ConsoleOne, SLP DA, and NICI from Windows servers.

IMPORTANT: Removing OpenText eDirectory also removes the roll-forward log directory and all the logs in it. If you want to be able to use the logs for restoring OpenText eDirectory on this server in the future, before removing OpenText eDirectory you must first copy the roll-forward logs to another location. For information about roll-forward logs, see "Using Roll-Forward Logs" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

- "Uninstalling OpenText eDirectory, ConsoleOne, and SLP DA" on page 175
- "Unattended Uninstallation of OpenText eDirectory" on page 176
- "Uninstalling NICI" on page 179
- "Uninstalling Microsoft Visual C++ 2005 and Visual C++ 2012 Runtime Libraries" on page 179

NOTE: The HTML files created using iMonitor are not removed. You must manually remove these files from `<install directory>\novell\NDS\ndsimon\dsreports` before removing eDirectory.

Uninstalling OpenText eDirectory, ConsoleOne, and SLP DA

1. On the Windows server where OpenText eDirectory is installed, click **Start** > **Settings** > **Control Panel** > **Add/Remove Programs**.
2. Select **OpenText eDirectory**, **ConsoleOne**, or the **SLP Directory Agent** from the list, then click **Add/Remove**.
3. Confirm that you want to remove your selection by clicking **Yes**.

   The Installation Wizard removes the program from the server.

Unattended Uninstallation of OpenText eDirectory

On Windows, the unattended uninstallation of OpenText eDirectory uses predefined text files that facilitate the unattended uninstallation.You can perform the following actions by using the unattended uninstallation mode of OpenText eDirectory:

- Deconfiguration of the installed OpenText eDirectory.
- Standalone uninstallation of OpenText eDirectory.
- Both uninstallation and deconfiguration of eDirectory.

The following sections discuss various features of unattended OpenText eDirectory uninstallation:

- "Response Files" on page 176
- "remove.rsp File Sections and Keys" on page 176
- "Add Features to the Automated Uninstallation" on page 177
- "Remove Configuration File Changes" on page 178
- "Unattended Uninstallation of OpenText eDirectory using Response File" on page 178

Response Files

Uninstalling OpenText eDirectory on Windows operating system can be made silent and more flexible by using a response file (`remove.rsp`) to complete the following tasks:

- Complete unattended uninstallation with all required user inputs
- Default configuration of components
- Bypass all prompts during the installation

A response file is a text file containing sections and keys, similar to a `Windows.ini` file. You can create and edit a response file by using any ASCII text editor. The OpenText eDirectory reads the uninstallation parameters directly from the response file and replaces the default uninstallation values with response file values. The uninstallation program accepts the values from the response file and continues to uninstall without prompts.

remove.rsp File Sections and Keys

The OpenText eDirectory uninstallation requires changes to the sections in the response file to add information about including the tree name, administrator context, administrator credentials (including user name and passwords), etc. A full list of the keys and their default values is available in the sample `remove.rsp` file that is delivered with the OpenText eDirectory installation.

NOTE: You should use the provided `remove.rsp` file available at `eDirectory\windows\x64\NDSonNT\remove.rsp` in the OpenText eDirectory installation. Essential parameters are set by default in this file. When editing the `remove.rsp` file, ensure there are no blank spaces between the key and the values along with the equals sign ("=") in each key-value pair.

You provide the administrator user credentials in the `remove.rsp` file for an unattended uninstallation. Therefore, you must permanently delete the file after the uninstallation to prevent the administrator credentials from being compromised.

Add Features to the Automated Uninstallation

Most details for configuring the OpenText eDirectory Uninstaller have default setting for the manual uninstallation. However, during unattended uninstallation, each configuration parameter must be explicitly configured. This section discusses the basic settings to be unconfigured.

OpenText eDirectory Server Details

The details of the server being uninstalled must be provided to the Uninstaller. Most of this information is configured in three tags, `[Novell:NDSforNT:1.0.0]`, `[Initialization]`, and `[Selected Nodes]`.

Take all the values mentioned in `[Initialization]` and `[Selected Nodes]` in `remove.rsp` as it they are.

[Novell:NDSforNT:1.0.0]

Tree Name: The name of the tree from which the server will be uninstalled.

Admin Login Name: The name (RDN) of the Administrator object in the tree that has full rights, at least to the context to which this server is added. All operations in the tree will be performed as this user.

Admin Context: Any user added to a tree has a user object that contains all the user-specific details. This parameter is the container object in the tree to which the Administrator object will be added. For primary server installations, this container will be created with the server object.

Admin Password: The password for the Administrator object created in the previous parameters. This password will be configured to the Administrator object during primary server installations. For secondary server installations, this needs to be the password of the Administrator object in the primary server that has rights to the context to which the new server is added.

NDS Location: The OpenText eDirectory install location in the local system where the libraries and binaries are copied. By default, OpenText eDirectory is installed into `C:\Novell\NDS` unless it is changed in the response file.

DataDir: Until OpenText eDirectory version 9.2, the DIB was installed inside the NDS location as a subfolder. Later, administrators were given the option to provide a different DIB location, because there might be too much data stored in the DIB to fit into the NDS location. Currently, by default the DIB is installed in the `Files` subfolder inside the NDS location, but administrators can change this parameter and provide a different location

mode: The type of setup on OpenText eDirectory. The three types of setup are:

- deconfigure: Performs the deconfiguration of OpenText eDirectory.
- uninstall: Performs uninstallation of OpenText eDirectory.
- full: Performs both deconfiguration and uninstallation of OpenText eDirectory.

NOTE: If you opt for the full setup mode during unattended install, then while uninstalling OpenText eDirectory you cannot opt for individual deconfiguration and uninstallation option.

ConfigurationMode: If the setup mentioned in the mode key is deconfigure, then ensure that you do not change the `RestrictNodeRemove` value of the `ConfigurationMode` key

Prompt: The type of the uninstallation mode should be mentioned in this variable. It will be set by default to 'silent' for unattended uninstallation. If any value other than 'silent' is set then it will do normal uninstallation

The following is a sample of text in the response file for all the basic parameters described above:

```
[Novell:NDSforNT:1.0.0]

  Tree Name=SILENTCORP-TREE

  Admin Context=Novell

  Admin Login Name=Admin

  Admin Password=novell

  prompt=silent
```

Remove Configuration File Changes

In the `remove.cfg` file residing in <*Windows Install Drive*>`\Program Files\Common Files\novell\ni\bin`, change

`[PARAMETERS]0/OUTPUT_TO_FILE`

to

`[PARAMETERS]0/OUTPUT_TO_FILE /SILENT`

Unattended Uninstallation of OpenText eDirectory using Response File

Copy the above edited file `remove.rsp` into <*Windows Install Drive*>`\Program Files\Common Files\novell\ni\data`.

The `install.exe` installed in the eDirectory is invoked in the command line with a few additional parameters. Depending on the required setup, you must use either of the following commands:

Deconfigure

```
<Windows Installed Drive>\Program Files\Common
Files\novell\ni\bin>install.exe -remove /restrictnoderemove /nopleasewait
..\data\ip.db ..\data\remove.rsp Novell:NDSForNT:1.0.0 0 NDSonNT
```

Uninstall

1. Rename the `ip.db` file present in the <*Windows Drive*>`\Program Files\Common Files\novell\ni\data` directory to another name.
2. Copy the `ip_conf.db` file in the <*Windows Drive*>`\Program Files\Common Files\novell\ni\data` folder to `ip.db`.
3. Run the following command:

   ```
   <Windows Installed Drive>\Program Files\Common
   Files\novell\ni\bin>install.exe -remove /nopleasewait ..\data\ip.db
   ..\data\remove.rsp Novell:NDSForNT:1.0.0 0 NDSonNT
   ```

Deconfiguration and Uninstallation of OpenText eDirectory

```
<Windows Installed Drive>\Program Files\Common
Files\novell\ni\bin>install.exe -remove /nopleasewait ..\data\ip.db
..\data\remove.rsp Novell:NDSForNT:1.0.0 0 NDSonNT
```

After performing an uninstallation of OpenText eDirectory or combination setup, delete the following folders:

- `C:\Novell\NDS` (default location, or else from the OpenText eDirectory installed directory)
- `C:\Novell\NDS\Files` (default location, or else from the OpenText eDirectory DIB location)
- `<Windows Installed Drive>:\Program Files\Common Files\Novell\ni`
- `<Windows Installed Drive>:\Windows\system32\NDScpa.cpl`

Uninstalling NICI

1. On the Windows server where OpenText eDirectory is installed, click **Start** > **Settings** > **Control Panel** > **Add/Remove Programs**.
2. Select **NICI** from the list, then click **Add/Remove**.
3. Confirm that you want to remove NICI by clicking **Yes**.

   The Installation Wizard removes NICI from the server.

After uninstalling NICI, if you want to completely remove NICI from your system, delete the `C:\Windows\system32\novell\nici` (32-bit) and `C:\Windows\SysWOW64\novell\nici` (64-bit) subdirectory. You might need to take ownership of some of the files and directories to delete them.

WARNING: After the `nici` subdirectory has been removed, any data or information that was previously encrypted with NICI will be lost.

Uninstalling Microsoft Visual C++ 2005 and Visual C++ 2012 Runtime Libraries

If Microsoft Visual C++ 2005 and Visual C++ 2012 Runtime Libraries are not used by any other products other than OpenText eDirectory, uninstall them by using the following procedure:

1. Navigate to *Add/Remove Programs* or **Programs and Features** on the Windows server where eDirectory is installed.
2. Remove the following redistribution package:

   `Microsoft Visual C++ 2005 Redistributable` and `Microsoft Visual C++ 2012 Redistributable (x64)`

Uninstalling OpenText eDirectory on Linux

Use the `nds-uninstall` utility to uninstall OpenText eDirectory components from Linux computers. This utility uninstalls eDirectory from the local host. You must deconfigure OpenText eDirectory server before running `nds-uninstall`. Run `ndsconfig rm -a <admin FDN>` to remove the eDirectory server. This utility is available at `/opt/novell/eDirectory/sbin/nds-uninstall`.

Note that you must not run `ndsconfig rm` on an OES server.

IMPORTANT: Removing OpenText eDirectory also removes the roll-forward log directory and all the logs in it. If you want to be able to use the logs for restoring eDirectory on this server in the future, before removing OpenText eDirectory you must first copy the roll-forward logs to another location. For information about roll-forward logs, see "Using Roll-Forward Logs" in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.

1. Execute the `nds-uninstall` command.
2. Use the following syntax:

   ```
   nds-uninstall [-s][-h]
   ```

   If you do not provide the required parameters in the command line, the nds-install utility will prompt for the parameters.

Parameter	Description
`-h`	Displays the help strings.
`-s`	Removes the OpenText eDirectory packages and binaries even when instances are configured. However, this option does not remove the DIB directory and the NDS configuration file. **IMPORTANT:** Ensure that using this option is not affecting other services for a long period.

nds-uninstall does not uninstall the following packages:

Package	Reasons for Not Removing
`NICI` package	`NICI` could be used by any of the following: ◆ Any other product ◆ OpenText eDirectory installed in a custom location ◆ OpenText eDirectory installed by a non-root user
`NOVLsubag`	`NOVLsubag` could be used by any of the following: ◆ OpenText eDirectory installed in a custom location ◆ OpenText eDirectory installed by a non-root user

Unattended Uninstallation of OpenText eDirectory on Linux

1. Remove the instances of OpenText eDirectory:

   ```
   ndsconfig rm -a <user name> -w passwd -c
   ```

2. Use either of the following in the automated script for the de-configuration of OpenText eDirectory:

   Passing the password through environment variable: `ndsconfig rm -a <user name> -w env:<environment variable> -c`

   Passing the password through file: `ndsconfig rm -a <user name> -w file:<filename with absolute/relative path> -c`

3. (Optional) In case of multiple instances, run the following command for individual instances:

   ```
   ndsconfig rm -a <user name> -w passwd  --config-file <absolute path for configuration file>
   ```

   For example:

   ```
   ndsconfig rm -a admin.novell -w n -c
   ```

   ```
   ndsconfig rm -a admin.novell -w env:ADM_PASWD -c
   ```

   ```
   ndsconfig rm -a admin.novell -w file:/Builds/88SP8/adm_paswd -c
   ```

4. To uninstall the OpenText eDirectory packages, run the nds-uninstall script to remove the OpenText eDirectory packages:

   ```
   nds-uninstall -u
   ```

Caveats for Uninstalling OpenText eDirectory

When you uninstall OpenText eDirectory and install it again, the OpenText eDirectory server cannot be accessible to the other servers in the network. All the distributed operations such as synchronization and obituary processing do not take place on the partitions whose replicas are present in the OpenText eDirectory server. If this state persists for a while, it might impact all the servers and the processes running on them.

Avoid uninstalling a newer version of OpenText eDirectory and install an earlier version, because:

- Does not revert the schema related upgrades.
- OpenText eDirectory might not be functional if DIB is upgraded to the newer version.
- Removes all the existing configuration files, except for the `nds.conf`.

However, consider the following when you uninstall a newer version of OpenText eDirectory and install an earlier version:

- Upgrade the DIB to the newer version. Else, eDirectory might not be functional.
- Back up the existing configuration files, except for the `nds.conf`, and restore when OpenText eDirectory is installed again.
- Does not revert the schema related upgrades.

A Linux Packages for OpenText eDirectory

OpenText eDirectory includes a Linux package system, which is a collection of tools that simplify the installation and uninstallation of various OpenText eDirectory components. Packages contain makefiles that describe the requirements to build a certain component of OpenText eDirectory. Packages also include configuration files, utilities, libraries, daemons, and man pages that use the standard Linux tools installed with the OS.

The following table provides information about the Linux packages that are included with OpenText eDirectory.

NOTE: On Linux, all the packages are prefixed with *novell-* except eba. For example, NDSserv is novell-NDSserv.

Package	Description
NOVLice	Contains the NetIQ Import Convert Export utility and is dependent on the NOVLlmgnt, NOVLxis, and NLDAPbase packages.
NDSbase	Represents the Directory User Agent. This package is dependent on the NICI package. The NDSbase package contains the following: ◆ Authentication toolbox containing the RSA authentication needed for eDirectory ◆ Platform-independent system abstraction library, a library containing all the defined Directory User Agent functions, and the schema extension library ◆ Combined configuration utility and the Directory User Agent test utility ◆ OpenText eDirectory configuration file and manual pages
NDScommon	Contains the man pages for the OpenText eDirectory configuration file, install, and uninstall utilities. This package is dependent on the NDSbase package.
NDSmasv	Contains the libraries required for mandatory access control (MASV).

Package	Description
NDSserv	Contains all the binaries and libraries needed by the OpenText eDirectory Server. It also contains the utilities to manage the OpenText eDirectory Server on the system. This package is dependent on the NDSbase, NDScommon, NDSmasv, NLDAPsdk, NOVLpkia and NOVLpkit packages. The NDSserv package contains the following: ◆ NDS install library, FLAIM library, trace library, NDS library, LDAP server library, LDAP install library, index editor library, DNS library, merge library, and LDAP extension library for LDAP SDK ◆ OpenText eDirectory Server daemon ◆ Binary for DNS and a binary to load or unload LDAP ◆ The utility needed to create the MAC address, the utility to trace the server and change some of the global variables of the server, the utility to back up and restore OpenText eDirectory, and the utility to merge OpenText eDirectory trees ◆ Startup scripts for DNS, NDSD, and NLDAP ◆ Man pages
NDSimon	Contains the runtime libraries and utilities used to search and retrieve data from OpenText eDirectory services. This package is dependent on the NDSbase package.
NDSrepair	Contains the runtime libraries and the utility that corrects problems in the OpenText eDirectory database. This package is dependent on the NDSbase package.
NLDAPbase	Contains LDAP libraries, extensions to LDAP libraries, and the following LDAP tools: ◆ ldapdelete ◆ ldapmodify ◆ ldapmodrdn ◆ ldapsearch This package is dependent on the NLDAPsdk package.
NOVLnmas	Contains all the NMAS libraries and the nmasinst binaries needed for NMAS server. This package is dependent on the NICI and NDSmasv packages.
NLDAPsdk	Contains OpenText extensions to LDAP runtime and Security libraries (Client NICI).
NOVLsubag	Contains the runtime libraries and utilities for the OpenText eDirectory SNMP subagent. This package is dependent on the NICI, NDSbase, and NLDAPbase packages.
NOVLpkit	Provides PKI Services which do not require OpenText eDirectory. This package is dependent on the NICI and NLDAPsdk packages.
NOVLpkis	Provides PKI Server Service. This package is dependent on the NICI, NDSbase, and NLDAPsdk packages.
NOVLsnmp	The runtime libraries and utilities for SNMP. This package is dependent on the NICI package.

Package	Description
NDSdexvnt	Contains the library that manages events generated in OpenText eDirectory to other databases.
NOVLpkia	Provides PKI services. This package is dependent on the NICI, NDSbase, and NLDAPsdk packages.
NOVLembox	Provides the eMBox infrastructure and eMTools.
NOVLlmgnt	Contains runtime libraries for OpenText Language Management.
NOVLxis	Contains the runtime libraries for NetIQ XIS.
NOVLsas	Contains the OpenText SAS libraries.
NOVLntls	Contains OpenText TLS library. This package is identified as ntls on Linux.
NOVLldif2	Contains the OpenText Offline Bulkload utility and is dependent on the NDSbase, NDSserv, NOVLntls, NOVLlmgnt, and NICI packages.
NOVLncp	Contains the OpenText Encrypted NCP Services for Linux. This package is dependent on the NDScommon package.
novell-eba	Contains the libraries for supporting enhanced background authentication. This package is dependent on NICI, NDSbase, and NDSServ packages.

B OpenText eDirectory Health Checks

OpenText eDirectory 9.3 provides a diagnostic tool to help you determine whether your OpenText eDirectory health is safe. The primary use of this tool is to check if the health of the server is safe before upgrading.

OpenText health checks are run by default with every upgrade and they occur before the actual package upgrade. However, you can run the diagnostic tool, `ndscheck`, to do the health checks at anytime.

Need for Health Checks

In earlier releases of OpenText eDirectory, the upgrade did not check the health of the server before proceeding with the upgrade. If the health was unstable, the upgrade operation would fail and OpenText eDirectory would be in an inconsistent state. In some cases, you probably could not roll back to the pre-upgrade settings.

This new health check tool resolves this, letting you to ensure that your server is ready to upgrade.

Performing Health Checks

You can perform OpenText eDirectory health checks in two ways:

NOTE: You need administrative rights to run the health check utility.

- "With the Upgrade" on page 187
- "As a Standalone Utility" on page 188

With the Upgrade

The health checks are run by default every time you upgrade OpenText eDirectory.

Linux

Every time you upgrade, the health checks are run by default before the actual upgrade operation starts.

To skip the default health checks, you can use the `-j` option with `nds-install`.

Windows

The OpenText eDirectory health checks happen as part of the installation wizard. You can enable or disable the health checks when prompted to do so.

As a Standalone Utility

You can run the OpenText eDirectory health checks as a standalone utility anytime you want. The following table lists the health check utility names for each platform.

Table B-1 *Health Check Utilities*

Platform	Utility Name

Linux	`ndscheck`
	
	Syntax:
	
	`ndscheck [--help | -?] Display command usage`
	`ndscheck [--version | -v] Display version`
	`information ndscheck [-h <hostname port]>] [-a`
	`<admin FDN>] [-F <log file>] [-D] [-q] [--config-`
	`file <file name>]`

Windows	`ndscheck`
	
	Syntax:
	
	`ndscheck [--help | -?] Display command usage`
	`ndscheck [--version | -v] Display version`
	`information ndscheck [-h <hostname port]>] [-a`
	`<admin FDN>] [-F <log file>] [-D] [-q] [--config-`
	`file <file name>]`

Types of Health Checks

When you run the `ndscheck` utility or upgrade, the following types of health checks are done:

- Basic Server Health
- Partitions and Replica Health

When you run the `ndscheck` utility, the results are displayed on the screen and logged in `ndscheck.log`. For more information on log files, refer to “Log Files” on page 190.

If the health checks are done as part of the upgrade, you are either prompted to continue the upgrade process or the process is aborted, depending on the types of errors found (if any). Error types are described in “Categorization of Health” on page 189.

Basic Server Health

This is the first stage of the health check, where the health check utility checks for the following:

1. The OpenText eDirectory service is up. The DIB is open and able to read some basic tree information such as tree name.
2. The server is listening on the respective port numbers.

For LDAP, it gets the TCP and the SSL port numbers and checks if the server is listening on these ports.

Similarly, it gets the HTTP and HTTP secure port numbers and checks if the server is listening on these ports.

Partitions and Replica Health

After checking the basic server health, it then checks the partitions and replica health as follows:

1. Checks the health of the replicas of the locally held partitions.
2. Reads the replica ring of every partition held by the server and checks whether all servers in the replica ring are up and all the replicas are in the ON state.
3. Checks the time synchronization of all the servers in the replica ring, showing any time difference between the servers.

Categorization of Health

There are three possible categories of health, based on the errors found while checking the health of an OpenText eDirectory server:

- Normal (page 189)
- Warning (page 189)
- Critical (page 190)

The status of the health checks is logged into a log file. For more information, refer to “Log Files” on page 190.

Normal

All the health checks were successful and the server health is normal.

The upgrade proceeds without an interruption.

Warning

Minor errors were found while checking the server health.

If the health check is run as part of the upgrade, you are prompted to either abort or continue.

Warnings normally occur in the following scenarios:

- Server not listening on LDAP and HTTP ports (normal, secure, or both).
- Unable to contact any of the non-master servers in the replica ring.
- Servers in the replica ring are not in sync.

Critical

Critical errors were found while checking the OpenText eDirectory health.

If the health check is run as part of the OpenText eDirectory upgrade, the upgrade operation is aborted.

The critical state normally occurs in the following scenarios:

- Unable to read or open the DIB (might be locked or corrupt).
- Unable to contact all the servers in the replica ring.
- Locally held partitions are busy.
- Replica is not in the ON state.

Log Files

Every OpenText eDirectory health check operation, whether it is run with the upgrade or as a standalone utility, maintains the status of the health in a log file.

The content of the log file is similar to the messages displayed on the screen when the checks are happening.

The health check log file contains the following:

- Status of the health checks (normal, warning, or critical).
- URLs where possible solutions can be found.
  - Support forums (http://forums.novell.com/netiq/netiq-product-discussion-forums/edirectory/)
  - Troubleshooting Documentation (https://www.netiq.com/documentation/edir88/edir88tshoot/data/bookinfo.html)
  - Error Codes (http://www.novell.com/documentation/nwec/)
  - Patches (http://support.novell.com/patches.html)
  - Cool Solutions (http://www.novell.com/communities/coolsolutions/edirectory)

The following table gives the default log file location on various platforms:

Table B-2 *Health Check Log File Location*

Platform	Log Filename	Location
Linux	`ndscheck.log`	1. If you use the `-h` option, the `ndscheck.log` file is saved in the user's home directory. 2. If you use the `--config-file` option, the `ndscheck.log` file is saved in the server instance's log directory. You can also select an instance from the multiple instances list.
Windows	`nsdcheck.log`	The log file will be saved at *`install_directory`*`\novell nds\`. **NOTE:** *`install_directory`* is user specified.

C Configuring OpenSLP for OpenText eDirectory

This appendix provides information for network administrators on the proper configuration of OpenSLP for OpenText eDirectory installations without the Novell Client.

- "Service Location Protocol" on page 193
- "SLP Fundamentals" on page 193
- "Configuration Parameters" on page 195

Service Location Protocol

OpenSLP is an open-source implementation of the IETF Service Location Protocol Version 2.0 standard, which is documented in IETF Request-For-Comments (RFC) 2608 (http://www.ietf.org/rfc/rfc2608.txt?number=2608).

In addition to implementing the SLP v2 protocol, the interface provided by OpenSLP source code is an implementation of another IETF standard for programmatically accessing SLP functionality, documented in RFC 2614 (http://www.ietf.org/rfc/rfc2614.txt?number=2614).

To fully understand the workings of SLP, it is worth reading these documents and internalizing them. They are not necessarily light reading, but they are essential to the proper configuration of SLP on an intranet.

For more information on the OpenSLP project, see the OpenSLP (http://www.OpenSLP.org) Web site and the SourceForge (http://sourceforge.net/projects/openslp) Web site. The OpenSLP Web site provides several documents that contain valuable configuration tips. Many of these are incomplete at the time of this writing.

SLP Fundamentals

Service Location Protocol specifies three components:

- The user agent (UA)
- The service agent (SA)
- The directory agent (DA)

The user agent's job is to provide a programmatic interface for clients to query for services, and for services to advertise themselves. A user agent contacts a directory agent to query for registered services of a specified service class and within a specified scope.

The service agent's job is to provide persistent storage and maintenance points for local services that have registered themselves with SLP. The service agent essentially maintains an in-memory database of registered local services. In fact, a service cannot register with SLP unless a local SA is present.

Clients can discover services with only a UA library, but registration requires an SA, primarily because an SA must reassert the existence of registered services periodically in order to maintain the registration with listening directory agents.

The directory agent's job is to provide a long-term persistent cache for advertised services, and to provide a point of access for user agents to look up services. As a cache, the DA listens for SAs to advertise new services, and caches those notifications. Over a short time, a DA's cache will become more complete. Directory agents use an expiration algorithm to expire cache entries. When a directory agent comes up, it reads its cache from persistent storage (generally a hard drive), and then begins to expire entries according to the algorithm. When a new DA comes up, or when a cache has been deleted, the DA detects this condition and sends out a special notification to all listening SAs to dump their local databases so the DA can quickly build its cache.

In the absence of any directory agents, the UA will resort to a general multicast query that SAs can respond to, building a list of the requested services in much the same manner that DAs use to build their cache. The list of services returned by such a query is an incomplete and much more localized list than that provided by a DA, especially in the presence of multicast filtering, which is done by many network administrators, limiting broadcasts and multicasts to only the local subnet.

In summary, everything hinges on the directory agent that a user agent finds for a given scope.

OpenText Service Location Providers

The OpenText version of SLP takes certain liberties with the SLP standard in order to provide a more robust service advertising environment, but it does so at the expense of some scalability.

For example, in order to improve scalability for a service advertising framework, you can limit the number of packets that are broadcast or multicast on a subnet. The SLP specification manages this by imposing restrictions on service agents and user agents regarding directory agent queries. The first directory agent discovered that services the desired scope is the one that a service agent (and consequently, local user agents) will use for all future requests on that scope.

The OpenText SLP implementation actually scans all of the directory agents it knows about looking for query information. It assumes a 300-millisecond round trip time is too long, so it can scan 10 servers in about 3 to 5 seconds. This doesn't need to be done if SLP is configured correctly on the network, and OpenSLP assumes the network is in fact configured correctly for SLP traffic. OpenSLP's response timeout values are greater than that of OpenTexts SLP service provider, and it limits the number of directory agents to the first one that responds, whether or not that agent's information is accurate and complete.

User Agents

A user agent takes the physical form of a static or dynamic library that is linked into an application. It allows the application to query for SLP services.

User agents follow an algorithm to obtain the address of a directory agent to which queries will be sent. Once they obtain a DA address for a specified scope, they continue to use that address for that scope until it no longer responds, at which time they obtain another DA address for that scope. User agents locate a directory agent address for a specified scope by:

1. Checking to see if the socket handle on the current request is connected to a DA for the specified scope. If the request happens to be a multipart request, there may already be a cached connection present on the request.

2. Checking its local known DA cache for a DA matching the specified scope.
3. Checking with the local SA for a DA with the specified scope (and adding new addresses to the cache).
4. Querying DHCP for network-configured DA addresses that match the specified scope (and adding new addresses to the cache).
5. Multicasting a DA discovery request on a well-known port (and adding new addresses to the cache).

The specified scope is “default” if not specified. That is, if no scope is statically defined in the SLP configuration file, and no scope is specified in the query, then the scope used is the word “default”. It should also be noted that OpenText eDirectory never specifies a scope in its registrations. That’s not to say the scope always used with OpenText eDirectory is “default.” In fact, if there is a statically configured scope, that scope becomes the default scope for all local UA requests and SA registrations in the absence of a specified scope.

Service Agents

Service agents take the physical form of a separate process on the host machine. In the case of Windows, `slpd.exe` runs as a service on the local machine. User agents query the local service agent by sending messages to the loop-back address on a well-known port.

A service agent locates and caches directory agents and their supported scope list by sending a DA discovery request directly to potential DA addresses by:

1. Checking all statically configured DA addresses (and adding new ones to the SA’s known DA cache).
2. Requesting a list of DA’s and scopes from DHCP (and adding new ones to the SA’s known DA cache).
3. Multicasting a DA discovery request on a well-known port (and adding new ones to the SA’s known DA cache).
4. Receiving DA advertising packets that are periodically broadcast by DAs (and adding new ones to the SA’s known DA cache).

Since a user agent always queries the local service agent first, this is important, as the local service agent’s response will determine whether or not the user agent continues to the next stage of discovery (in this case DHCP-- see steps 3 and 4 in “User Agents” on page 194.).

Configuration Parameters

Certain configuration parameters in the `%systemroot%/slp.conf` file control DA discovery as well:

```
net.slp.useScopes = <comma delimited scope list>
net.slp.DAAddresses = <comma delimited address list>
net.slp.passiveDADetection = <"true" or "false">
net.slp.activeDADetection = <"true" or "false">
net.slp.DAActiveDiscoveryInterval = <0, 1, or a number of seconds>
```

The `useScopes` option indicates which scopes the SA will advertise into, and which scopes queries will be made to in the absence of a specific scope on the registration or query made by the service or client application. Because OpenText eDirectory always advertises into and queries from the default scope, this list will become the default scope list for all OpenText eDirectory registrations and queries.

The `DAAddresses` option is a comma-delimited list of dotted decimal IP addresses of DAs that should be preferred to all others. If this list of configured DAs does not support the scope of a registration or query, then SAs and UAs will resort to multicast DA discovery, unless such discovery is disabled.

The `passiveDADetection` option is True by default. Directory agents will periodically broadcast their existence on the subnet on a well-known port if configured to do so. These packets are termed DAAdvert packets. If this option is set to `False`, all broadcast DAAdvert packets are ignored by the SA.

The `activeDADetection` option is also True by default. This allows the SA to periodically broadcast a request for all DAs to respond with a directed DAAdvert packet. A directed packet is not broadcast, but sent directly to the SA in response to these requests. If this option is set to `False`, no periodic DA discovery request is broadcast by the SA.

The `DAActiveDiscoveryInterval` option is a try-state parameter. The default value is `1`, which is a special value meaning that the SA should only send out one DA discovery request upon initialization. Setting this option to `0` has the same effect as setting the `activeDADetection` option to `false`. Any other value is a number of seconds between discovery broadcasts.

These options, when used properly, can ensure an appropriate use of network bandwidth for service advertising. In fact, the default settings are designed to optimize scalability on an average network.

D Troubleshooting Issues

This section provides useful information for troubleshooting problems with installing and configuring OpenText eDirectory.

Troubleshooting the Installation Issues

The following table lists the issues you might encounter and the suggested actions for working on these issues. If the problem persists, contact your OpenText representative.

Issues	Suggested Actions
Installation Takes a Long Time. When you are installing OpenText eDirectory into an existing tree and the installation takes a long time to complete, look at the DSTrace screen on the server. If the `-625 Transport failure` message appears, you need to reset the address cache	To reset the address cache, enter the following command at the system console: `set dstrace = *A`
OpenText eDirectory Install Fails for Container Administrators The OpenText eDirectory 9.0 installation program supports installations by administrators who have supervisor rights to the container that the server resides in. In order to handle this, the first server that OpenText eDirectory 9.0 is installed into must have supervisor rights to [Root] to extend the schema. From that point on, subsequent servers do not have to have rights to [Root]. However, with OpenText eDirectory 9.0, depending on the platform that OpenText eDirectory 9.0 is installed in to first, all schema might not be extended, requiring supervisor rights to [Root] for subsequent server installations on different platforms.	If OpenText eDirectory 9.0 will be installed on multiple platforms, make sure that you have supervisor rights to [Root] for the first server OpenText eDirectory will be installed on for EACH platform. For example, if the first server that OpenText eDirectory 9.0 is going to be installed on is running Linux, and OpenText eDirectory 9.0 will also be installed on Solaris, the first server for each platform must have supervisor rights to [Root]. Subsequent servers on each platform will only have to have container administrator rights to the container where the server is being installed. For additional information, see solution NOVL83874 (http://support.novell.com/docs/Tids/Solutions/10073723.html) in the *OpenText eDirectory 8.7.x Readme Addendum*
Default Listeners for New Network Interface	On Windows, OpenText eDirectory listens on all interfaces configured on the computer for NCP, HTTP, HTTPS, LDAP and LDAPS by default. Adding a new network interface address to the computer, and restarting OpenText eDirectory will make it start listening on that address automatically, and have referrals also added correspondingly. **NOTE:** On Linux, we need to manually add interfaces to `n4u.server.interfaces` parameter.

Issues	Suggested Actions
Replication Issues After an Upgrade When you upgrade to OpenText eDirectory 9.0 and enable encrypted replication, replication fails in rare scenarios.	To workaround this issue: 1. In OpenText Identity Console, select **Object Management**. 2. On the **Objects** window > click the **Types** drop-down tab > select NCP Server > click **Search**. 3. Under **Search Results** > select the NCP Server 4. Under the General tab, select **Other**. 5. On the **Modify Objects** tab > click **Valued Attributes** . 6. On the SELECT ATTRIBUTES screen, select **NCPKeyMaterialName** > click **OK**. For example, **SSL CertificateDNS**. 7. Run Limber on the server where the attribute is changed (Step 6). For information about using Limber, see the OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide.
Error 1603 encountered during OpenText eDirectory patching Following `sysprep.exe` execution in Windows, installing or updating eDirectory to a directory requiring authenticated user group privileges may result in a 1603 error. The **Administrator** user was found to be part of both the **Authenticated Users Group** (all logged-in users) and the **Administrators Group**. According to Microsoft troubleshooting guidelines, this dual group membership can cause conflicts with the User Account Control (UAC) system, potentially blocking installation privileges for the administrator. Adjusting the privileges for the **Authenticated Users** group resolved the issue, allowing the upgrade to complete successfully.	To resolve installation/update failures of OpenText eDirectory on Windows systems after `sysprep.exe` execution, the administrator should modify the OpneText eDirectory installed directory permissions. Specifically, remove the **Authenticated Users** group from the directory's permissions. This can be achieved by navigating to the directory's **Properties** and accessing the **Security** tab, where the **Authenticated Users** group can be removed from the list of granted privileges.
When you upgrade eDirectory from version 9.2.x to 24.4(v9.3) and later, attributes with Integer 64 syntax show up as different values.	There is a workaround mentioned to resolve the issue. For more information, see After upgrade to eDirectory 9.2.9 (or higher) attributes with syntax Integer 64 show different values (https://portal.microfocus.com/s/article/KM000039161?language=en_US)

Issues	Suggested Actions
After upgrading eDirectory 9.2.7 to eDirectory 9.3, IDM is unable to retrieve distribution password for some users. IDM drivers report: Code(-9065) Unable to determine value of attribute nspmDistributionPassword for object (DN of a user object). This is due to the **56 bit/ DES keys** that exist in the tree.	From eDirectory 24.4(v9.3), 56-bit/ DES key configurations are no longer supported, with or without FIPS mode. This is due to OpenSSL 3.0, which supports stronger key configurations. For guidance on identifying and migrating away from 56-bit keys, refer to the following TID: KM000040457 (https://portal.microfocus.com/s/article/KM000040457?language=en_US).

Troubleshooting the Configuration Issues

The following table lists the issues you might encounter and the suggested actions for working on these issues. If the problem persists, contact your OpenText representative.

Issues	Suggested Actions
Loopback Referrals Are Returned By a Directory Server	When OpenText eDirectory is configured to listen on loopback addresses, the loopback addresses are stored and returned to the clients when they perform searches and other operations. The referrals are not applicable to the clients that attempts to connect from the machines other than the server. Therefore, the clients fail to connect by using those loopback referrals. However, the other referrals returned by the server still work for the clients. Trying to connect to each loopback referrals and then choosing the correct referrals could affect the performance of the clients.

Tree Name Lookup Failed: -632 Error While Configuring OpenText eDirectory 9.0 on Linux	While configuring OpenText eDirectory 9.0 on Linux, you might get the Tree name lookup failed: -632 error. To resolve this, perform the following steps:
	
	1. After installing the SLP package, ensure that you manually start SLP as follows:
	
	`/etc/init.d/slpuasa start`
	
	2. After uninstalling the SLP package, ensure that you manually stop SLP as follows:
	
	`/etc/init.d/slpuasa stop`

Adding an EBA Enabled Secondary Server to a non-EBA Enabled Server Results in Configuration Failure	To workaround this issue, first configure the secondary server without EBA settings, and then upgrade to EBA using the EBA configuration settings.

Issues	Suggested Actions
Excluding the DIB directory from Backup or Antivirus Processes After installing OpenText eDirectory, you should configure your environment to exclude the DIB directory on your OpenText eDirectory server from any antivirus or backup software processes. If you do not exclude the DIB directory from processes of this type, you may encounter corrupted DIB files or `-618 FFFFFD96 INCONSISTENT DATABASE` errors.	Use the OpenText eDirectory Backup Tool to back up your DIB directory. For more information about backing up OpenText eDirectory, see Backing Up and Removing Roll-Forward Logs in the *OpenText™ eDirectory CE 24.4 (v9.3) Administration Guide*.
IP AG Certificate Does Not Get Created on SLES 11 64-Bit Platform	Consider a scenario where OpenText eDirectory 9.0 has both IPv4 and IPv6 configured and only one of the them (for example, IPv4) has an entry in the `/etc/hosts` file, and the other interface is accessible from a remote machine. If you configure OpenText eDirectory to listen on both the IPs, the IP AG certificate is generated only for the IP that is listed in the `/etc/hosts` file. In this example, it is generated for IPv4
Default Instance Path for Multiple Instances. While you configure the second instance of OpenText eDirectory on your host, you are prompted for the default path.	Select a different path and proceed.
ndsconfig Utility Displays Error While Configuring eDirectory as Non-root User ◆ `Group edirAdmin set failed` ◆ `WARNING: Unable to set permission on directory` ◆ `Checking if server is ready to service requests` A user might encounter these errors while installing or upgrading OpenText eDirectory 9.2.5 or later version, if any the following conditions is true: ◆ If the edirAdmin user group is missing from OpenText eDirectory. ◆ The user is not a member of the edirAdmin user group.	To troubleshoot this issue, the administrator must first create the edirAdmin user group manually, then add the user to this group. On the server where OpenText eDirectory is installed, open a terminal and run the following commands: ◆ To add the edirAdmin group, run `groupadd edirAdmin` ◆ To add a user to the edirAdmin group, run usermod -a -G edirAdmin <username> Where <`username`> is the name of the user. For example, to add the user John to the edirAdmin group, you must execute the command: `$ usermod -a -G edirAdmin john` **NOTE:** Before adding the user to the edirAdmin user group, the administrator must ensure that the user is not logged in through any terminal session. After adding, the user can login and configure OpenText eDirectory using the ndsconfig utility.

Troubleshooting the Issues with Multiple Instances of OpenText eDirectory

The following table lists the issues you might encounter and the suggested actions for working on these issues. If the problem persists, contact your OpenText representative.

Issues	Suggested Actions
If the First Instance is Down, HTTP Does Not Work On Linux, if OpenText eDirectory is configured on a box with multiple NIC cards and if HTTP is bound to more than one interface; if the first interface goes down, HTTP would not be accessible from the remaining interfaces. This is because the remaining interfaces will redirect the request to the first one, but the first interface is down.	To resolve this issue, if the first interface goes down, restart OpenText eDirectory.
ndsd Falls Back to Default Port if the Interface Specified is Incorrect	When using `ndsconfig new` or `ndsmanage` to create a second instance of the directory, if the interface specified is incorrect, nds tries to use the default interface. If you specify non default port (for example 1524), the interface specified is incorrect, it uses the default interface and the default port of 524. For `n4u.server.interfaces`, if the interface specified is incorrect, then `ndsd` will try to listen on the first interface and the port number would be the one specified in `n4u.server.tcp-port`.

How to Rebuild .edir Directory	The `.edir` directory is used for tracking multiple instances of OpenText eDirectory. To recreate the lost or corrupted instances file (`instances.$uid` file, where `$uid` specifies the user ID of the user in the system), you must create its individual instances file.
	
	These files must contain the absolute location of the `nds.conf` files of all the instances configured by the user. For example, a user with `uid` as 1000 must create an `/etc/opt/novell/eDirectory/conf/.edir/instances.1000` instances file with the following entries:
	
	`/home/user1/instance1/nds.conf`
	
	`/home/user1/instance2/nds.conf`

ndsconfig Utility

The following table lists the issues you might encounter and the suggested actions for working on these issues. If the problem persists, contact your OpenText representative.

Issues	Suggested Actions
Configuring ndsconfig to Run From Non-Default Location	If you receive an error when running the ndsconfig utility from a location other than the default `/opt/novell/eDirectory/bin` directory, ensure that you export the ndspath before running ndsconfig. Use the following command: `source /opt/novell/eDirectory/bin/ndspath` After you export the command, enter `ndsconfig` to run the ndsconfig utility, instead of `./ndsconfig`.
ndsconfig Does Not Verify an Invalid Configuration File Path	To create the necessary configuration file, ndsconfig requires the full path and the configuration filename. When the same path name is passed for both the configuration file and the instance directory, ndsconfig cannot create the configuration file and aborts the operation.
ndsconfig get Outputs Junk Characters for Non-English Characters	The `ndsconfig get` command outputs junk characters on Linux for some parameters that contain non-English characters. To work around this, enter the specify parameter name you want to get, as follows: `ndsconfig get <parameter_to_be_displayed>` For a list of parameters, refer to the `nds.conf` man page.
The `NDSD_NLDAP_IGNORE_CRITICALITY = True` environment variable is missing from the `/etc/opt/novell/eDirectory/conf/env` file while upgrading OpenText eDirectory on OES 2018SP2.	To troubleshoot this issue, you must manually run the ndsconfig upgrade command to add the `NDSD_NLDAP_IGNORE_CRITICALITY = True` environment variable in the `/etc/opt/novell/eDirectory/conf/env` file.

Troubleshooting NMAS Installation

- If you uninstall the Novell Client, you must uninstall and reinstall the NMAS Client if it is used by another application.
- You must have NMAS installed on a server that holds a writable replica of the user's object in order for the user to use NMAS.
- You must have the Novell International Cryptographic Infrastructure (NICI) Client installed on each client workstation that will run the NMAS software.

- If you do not restart the server after installing NMAS and you try to reset passwords, you receive an error message.
- You should keep the login method up to date. The OpenText eDirectory OES/Linux installs might not provide a way to upgrade the method.

Troubleshooting Certificate Server Installation

File Data Conflict During Installation

If you receive a message indicating that a newer file exists from the previous installation, you should select to always overwrite the newer file.

Incomplete List of Servers

The list of servers shown during the installation might not list servers that are configured to use only IP. You can install OpenText Certificate Server on a server whose name is not listed by typing the name of the server in the text box.

Failures During Installation

If the installation fails during the creation of the Organizational CA or the server certificate, or during the exportation of the trusted root certificate, the installation doesn't need to be repeated. The software has been successfully installed at this point. You can use Identity Console to create an Organizational CA and server certificates and export the trusted root.

IP Auto Generated Certificate Is Not Created on SLES 11 64-Bit Platform

Consider a scenario where OpenText eDirectory 9.0 has both IPv4 and IPv6 configured and only one of the them (for example, IPv4) has an entry in the /etc/hosts file, and the other interface is accessible from a remote machine. If you configure OpenText eDirectory to listen on both the IPs, the IP AG certificate is generated only for the IP that is listed in the /etc/hosts file. In this example, it is generated for IPv4.

IP Auto Generated IPv6 Certificate is Not Created When the Length of the Certificate Object RDN Exceeds the Maximum Limit

While installing OpenText eDirectory 9.0, which is listening on both IP v4 and IPv6 addresses, IP AG <IPv6> certificate (KMO) is not created.

This occurs when the length of the RDN of the certificate object exceeds the maximum limit of 64 characters. To handle this, a compressed format of IPv6 address is used so that even if the length exceeds the maximum limit, the address is split to accommodate the request. The address is split from the third colon (from the reverse order) in the address.

For example, if the IPv6 address is 2508:f0g0:1003:0061:0000:0000:0000:0002, then the truncated address is 0000:0000:0002. This ensures that the host is identifiable even after the address is truncated.

HTTP Server Associates With the IP AG Certificate When the Default Server Certificates are Recreated for a Server where CA is not Hosted

Use Identity Console to manually change the default association. The following procedure explains how to change:

1. Log in to Identity Console > **Object Management** > from the **Type** drop-down menu > select http server object > click **Search**.
2. On the **Search Results** menu > select the http server - <server name/CN>.
3. On the **Modify Objects** tab > click **Valued Attributes** .
4. On the **SELECT ATTRIBUTES** screen > search for httpKeyMaterialObjects attribute > click **Edit** .
5. Edit the HTTP server object association to SSL Certificate DNS > click **OK**.

Troubleshooting Installation

If bridge patch is not applied on EBA enabled OpenText eDirectory 9.2.8 environment, adding OpenText eDirectory 9.3 replica throws the following error messages for -

- **Linux:** One or more EBA enabled servers in the ring, does not have the bridge patch applied.
- **Windows:** Credentials are incorrect, please verify and retry

Workaround: Apply the bridge patch to all the servers.